OPPORTUNISTIC RESOURCE MANAGEMENT TO IMPROVE NETWORK SERVICE PERFORMANCE IN USER-CREATED NETWORKS

BY

NATHANAEL A. THOMPSON

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Doctoral Committee:

Associate Professor Robin Kravets, Chair
Associate Professor Tarek Abdelzaher
Associate Professor Indranil Gupta
Petros Zerfos, Ph.D.

Abstract

Internet service providers have always struggled to provide sufficient last-hop connectivity to end-users. Even as new broadband technologies are beginning to meet end-user demands for indoor access, end-users are now demanding always-on high-speed mobile connectivity where ever they are. Wireless network operators are challenged by limited wireless resource availability that prevents mobile broadband speeds similar to the levels enjoyed at indoor static locations. However, as the use of wireless technology has increased, the availability and proliferation of Wi-Fi enabled devices has exploded to the point that ample idle bandwidth, storage and computation resources are available in end-user owned and operated devices. By collaboratively pooling these idle resources into user-created networks, the network capacity to mobile end-users can be greatly increased. Doing so is challenging because of the dynamic and unreliable availability of end-user resources.

The thesis of this work is that in spite of the dynamic and disconnected nature of user-created networks, high-throughput, high-availability and low-latency network services for mobile end-users can be enabled through localized and location-based solutions using only end-user contributed resources. This is demonstrated through the design and implementation of four services including i) node-based local congestion control that improves message delivery rate for store-carry-and-forward routing in disconnected networks, ii) a location-based data overlay running on top of mobile devices, iii) localized flow scheduling to increase bandwidth for mobile clients connecting to external networks and iv) local, distributed authentication with location-dependent certificate revocation list segments to reduce authentication delays. These services are evaluated through deployments in real testbeds and through simulations of large-scale mobile networks.

Dedicated to my family: Ramona, Sean and Jonas.

Acknowledgments

In all your ways acknowledge the Lord, and he will make your paths straight.

Many thanks to my advisor, Professor Robin Kravets for taking me in on short notice and for the valuable advice and direction she gave me while finishing my dissertation. Thank you also to Professor Indranil Gupta who always demonstrated a clear ideal for doing good research, Petros Zerfos for guiding me through a tough period beyond his duties and Professor Tarek Abdelzaher for his creative ideas.

Thank you to my fellow peers who have taught me and encouraged me: Jay Patel, Steve Ko, Ramsés Morales, Sam Nelson, Mehedi Bakht, Riccardo Crepaldi, Farhana Ashraf, Chun-cheng Chen, Eunsoo Seo and Dawid Piwinski.

Finally, I owe much of my perseverance to my wife Ramona who always kept my thoughts positive. She has been a blessing as a friend, wife and mother.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Ubiquitous data access at high speed and low cost has been a long-standing vision for years, yet it is one Internet service providers (ISPs) have struggled to meet. Users are continuously growing more dependent on instantaneous data access for work, entertainment and their daily routines. Their demands for data connectivity continually push the limits of existing last-mile network capacity. While sufficient indoor Internet connectivity is becoming nearly pervasive in the U.S. as a result of recent fiber-to-the-home and improved cable broadband networks, there is additionally a growing demand for global mobile data access both indoors and out using wireless technologies. Fulfilling this demand is difficult due to the limited resources available for wireless communication.

The dominant approach in today's market for provisioning mobile data access is through cellular networks. 3G and emerging 4G networks have proven popular due to their good coverage and usually reliable connectivity. The strength of these networks is that they operate in a very controlled environment. Basestation deployment is planned for the best coverage, dedicated backhaul provisions each access point and licensed spectrum ensures minimal interference in the wireless channel. However, providing such a controlled networking environment comes with drawbacks. As the capacity limit of these networks is reached, growing the network for more capacity is both slow and expensive as it requires extensive planning, difficult installations, and expensive spectrum licensing fees. As demonstrated by the release of Apple's iPhone, which increased data traffic 5000% [59], data demands are growing faster than cellular networks alone can handle.

Alternative wireless data access networks could alleviate the pressure on the cellular infrastructure by off-loading much of the end-user traffic. However, existing solutions for wide-area wireless access all suffer problems that prevent their widescale deployment. Wireless mesh networks [4, 9, 12], touted as a low-cost last-mile alternative for broadband Internet access, suffer intra- and inter-flow interference and wireless channel dynamics that limit their scale [66]. They provide acceptable performance only at distances limited to a small number of hops. In addition, to achieve acceptable coverage, there is a high cost to install infrastructure and maintain the necessary Wi-Fi access points. Public Wi-Fi hotspots, usually deployed

in popular areas such as airports, malls and cafes [2, 14] are even less appropriate for providing wide-scale mobile data access. Such networks provide very limited geographic coverage and bear a higher cost of installation that includes installing backhaul to each access point on top of infrastructure deployment and maintenance costs. The cost of these hotspot networks impedes further expansion of the network. Government or industry-backed solutions have to this point seen only limited success due to the lack of proven business models for regaining the installation and maintenance costs. With all of the above approaches, growing network capacity depends on resources that are only available through costly infrastructure deployments.

However, an abundance of low-cost resources already exist that can be used to provide mobile end-user networking, if they can be harnessed. The reduced cost and improved efficiency of Wi-Fi communication has led to the scenario where millions of Wi-Fi enabled communication devices are carried and deployed by end-users. These devices are often equipped with excess bandwidth, storage and computation resources that are idle most of the time, creating a potential utopia for on-demand networking. By collaboratively pooling these idle resources and carrying each other's traffic, the network capacity to mobile end-users can be greatly increased. Because these user-created networks (UCNs) are deployed by end-users themselves, the monetary cost to expand network coverage is close to zero. In addition, these networks are formed where the users are, extending network coverage to areas where infrastructure might not be available.

1.1 Required Services for User-created Networks

If such UCNs can be successfully deployed alongside existing networks, the vast resources and new coverage possibilities can greatly advance the state of ubiquitous mobile data access. However, because the devices that form the network are owned and operated by end-users, the UCN is an environment unique from traditional networks that requires a large suite of services to make it a reality. Primary amongst all services are those that enable communication within the network. Incentive management, privacy and security, business models, automation and configuration tools, etc. all depend on the underlying networking functionality. As such, the focus of this work is on designing high-performance networking services for user-created networks that efficiently satisfy the communication needs of the network.

The set of networking services can be divided into two categories: services for enabling intra-network communication and those for inter-network communication. First, services must be developed to provide intra-network connectivity between users and between users and data. Intra-network services make the UCN a viable self-contained network. Second, there must be services to enable inter-network communi-

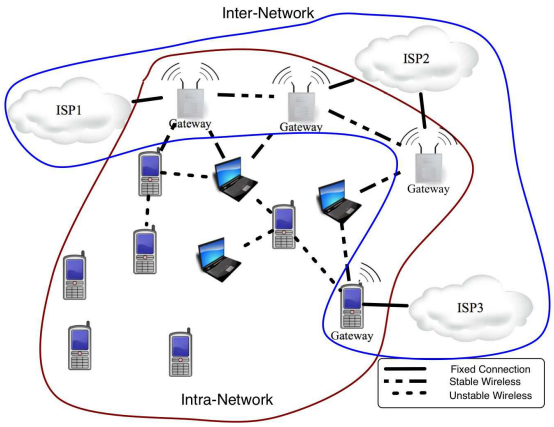


Figure 1.1: Communication in a user-created network includes areas of intermittent connectivity, ad hoc connections, static mesh links and fixed broadband connections. Nodes with Internet connections can become gateways for the network.

cation connecting the UCN to external networks. It is unlikely that the UCN will ever be rich enough to satisfy all of the end-user networking needs by itself. Much as existing stub networks are connected to the Internet through external ISPs, so too must inter-network communication with the UCN be enabled.

1.2 Networking Challenges

The difficulty in deploying these services for the UCN lies in the unique nature of a network formed entirely from end-user devices. The user-created network is composed of the devices and the network links that arise as the devices come in and out of range of one another (see Figure 1.1). The number of network links between devices fluctuates as the density of users changes from human and device mobility [47]. This mobility along with fluctuations in the wireless channel [21], unpredictable node membership [33] and unreliable service, leads to dynamic connectivity. The network deployment itself is unplanned, forming spontaneously when possible. All of these dynamics combine to result in frequent periods of disconnectivity [52, 110, 133], which requires new modes of operating for supporting services.

Any solutions to overcome the dynamic connectivity in the UCN cannot rely on supporting infrastructure to achieve their goal. The user-created network is deployed by end-users who only manage their own set of devices at the edge of existing networks and typically are not able to deploy services in the Internet. If Internet-based services were to be used, they would have to be deployed by a capable third party or ISP. However, such a system would have to cover the potentially global deployment of the UCN and scale to millions of users. Given the mindset of users in a shared, user-generated network, it is unlikely that those users would be willing to pay even a small amount to use such a service, making it hard to find effective business models for third parties to recoup their costs of running a service. Additionally, any in-

frastructure that does exist will be unreachable for extended periods of time due to the intermittent connectivity and unmanaged connections to the Internet. Given these challenges, decentralized solutions are needed for any service that runs in the UCN.

Finally, the network is formed entirely from end-user devices that have limited resources including bandwidth, storage, computation, and energy. Carrying traffic for the UCN is a secondary task for all of the devices and so what limited resources are available must be managed carefully. Therefore, any service deployed in the UCN must have complementary resource management that maximizes performance under limited resource availability. Services must operate opportunistically, able to take advantage of resources when they are available, but also able to operate effectively when resources are limited.

Providing intra- and inter-network communication in the UCN is made difficult by the network dynamics within UCNs. Specifically, intra-network communication is challenged by the intermittent connectivity that arises from node mobility. Due to this intermittent connectivity, any communication within the network must be disruption-tolerant [55], opportunistically using whatever node contacts and storage are available to find new forwarding paths through the network. However, nodes have limited buffer space and contact duration that must be effectively managed for acceptable delivery rates. Inter-network communication is similarly challenged in the UCN by the lack of an authoritative network operator. Because there is no operator, the UCN is very unpredictable and unmanaged and in conflict with the controlled and predictable ISP networks to which it connects. New management approaches are needed to protect external resources yet enable adequate performance for internal clients. These management services must opportunistically use the available connectivity and information to satisfy both external network operators and UCN clients while managing the limited bandwidth and computation on each device.

1.3 Contributions

While some of these requirements have been studied before, existing research has focused on targeted domains and is largely inapplicable to the user-created network environment. Existing approaches that are centralized, global and/or static will not be able to run effectively in the UCN. The thesis of this work is that in spite of the dynamic and disconnected nature of user-created networks, high-throughput, high-availability and low-latency network services for mobile end-users can be enabled through localized and location-based solutions using only end-user contributed resources. This is demonstrated through the development of four services for both intra- and inter-network communication. These services include intra-network i) data forwarding and ii) data overlay as well as inter-network iii) connectivity management and

iv) authentication. Each of these services is built on decentralized, local and dynamic algorithms and protocols combined with opportunistic resource management techniques that enable high-throughput, low-latency and reliable services for user-created networks. The performance of each of the services is verified through deployments in real testbeds and through simulation of large-scale mobile networks.

1.4 Thesis Outline

Chapter 2 describes the specific challenges to deploying intra-network communication services in the UCN that arise from the intermittent connectivity experienced by end-user devices. In response to that connectivity challenge, Chapter 3 describes a new node-based local congestion control algorithm for replication-based store-carry-and-forward data forwarding within UCNs. The algorithm results from a Markovian model of message delivery in intermittently-connected networks. Having provided higher delivery rates for user-to-user communication, Chapter 4 connects users to data through a location-based, disruption-tolerant data overlay on top of mobile devices in the UCN. The Locus overlay stores data in the network at the data's home location. Location-based forwarding and dropping policies ensure a high lifetime for data and high probability query service. Chapter 5 describes in detail the challenges of managing inter-network communication. Building on a network model in which individual users manage the connectivity between external networks and the UCN, the PERM system in Chapter 6 aggregates the limited available bandwidth to improve the throughput and latency of inter-network communication for UCN clients. PERM is based on predictive techniques that take advantage of local information about individual end-users. Once inter-network connectivity for the UCN is managed, access control must also be addressed. Chapter 7 describes AGE, a decentralized, local authentication framework for access control in the UCN. To reduce the computation and storage requirements on each device, authentication material is segmented based on a user's home location. Devices opportunistically authenticate each other using as much information as they have available. All of these services provide the networking foundation needed to grow user-created networks. The work ends with future directions for UCNs in Chapter 8 and the conclusion in Chapter 9.

Chapter 2

Intra-network Communication

Intra-network communication is the foundation of any network. It allows the entities in the network to be connected using only the internal resources of the network. Traditionally, the focus has been on providing end-to-end message forwarding between two devices. While message forwarding remains a key service, a new focus is emerging for designing network services. Given the large amounts of data consumed and also *generated* by end-users, network design is starting to focus on the higher level challenge of connecting users to data objects irrespective of the underlying devices supporting the communication. Typically, both of these services are provided through structured systems such as Internet routing, databases or distributed hashtables.

Structured approaches work well in environments where the network nodes are stable and available and where communication is mostly reliable. However, the dynamics of the user-created network leads to unstable connectivity arising from mobility [47], fluctuations in the wireless channel [21], unpredictable node membership [33] and unreliable service. The network is unplanned and nodes may be highly mobile leading to unprecedented churn in node availability. These dynamics lead to frequent periods of disconnectivity between nodes [52, 110, 133]. This intermittent connectivity is the primary challenge for deploying services for intra-network communication in UCNs. Intermittent connectivity results in a lack of instantaneous end-to-end paths in the network, breaking existing approaches. Because of this, services deployed in the UCN must be disruption-tolerant [55], requiring new approaches for both message forwarding and data services in UCNs.

2.1 Message Forwarding

In most existing networks, message forwarding is done by first finding a route from the sender to the receiver through the connections between devices in the network, and then forwarding data along the discovered path. Many protocols for wired networks assume the set of potential network nodes is static and therefore routes change slowly over time only as network loads change or failures occur [103]. If, however,

wireless connections are used, the link conditions change much faster and therefore new mesh routing protocols were developed that change routes much more frequently [9, 12]. Routing between mobile nodes is even more difficult as the set of neighboring nodes is constantly changing. To handle mobility, routes must be constantly re-discovered to update for the changing connectivity [79, 109].

While successful in their relative domains, all of these protocols assume that an instantaneous path between nodes exists and if none can be found, a failure occurs. On the other hand, in intermittently-connected network (ICN) environments like the UCN, the opposite is true. Instantaneous end-to-end paths are rarely, if ever available. Instead paths only become available over time as nodes move into contact range of other nodes that can complete the path. Delivering messages requires finding routes across both space and *time*.

To handle the intermittent connectivity in the UCN and achieve higher delivery rates, a new approach for data forwarding is needed that forwards in both the spatial and temporal domains. In response, a new forwarding paradigm for ICNs has been developed called store-carry-and-forward (SCF) [137]. Under SCF routing, nodes hold messages in the local buffer and carry them until new contacts occur. During each contact the nodes opportunistically forward locally stored messages to the new peer if the node is suitable to carry the messages. Depending on the amount of information available to the nodes in the network, SCF routing may only be probabilistic, dependent upon the contacts that occur in the network. Many protocols have been developed based on the SCF approach with different optimizations to improve the message delivery probability [30, 37, 53, 93, 105, 121, 128]. One popular technique to increase the message delivery is multi-copy routing in which messages are replicated between nodes. Instead of forwarding only a single copy of each message in the network, multiple message copies are made, which allows the network to explore multiple paths simultaneously improving the chance that at least one copy reaches the destination.

Although message replication can improve network-wide delivery rates, this technique puts an extra burden on nodes' limited storage resources. Because a message is potentially copied every time a node contact occurs, the total storage requirements in the network can quickly overwhelm the network's capacity. As the rate at which messages are generated increases, if replication is too high, the network will become congested, dropping messages from buffers too soon before they have time to reach the destination, and actually reducing delivery rates. On the other hand, under-replicating potentially misses delivery opportunities, also reducing the network-wide message delivery rate. To manage this trade-off, congestion control is needed to adapt the replication rate to maximize delivery.

Existing congestion control mechanisms in the Internet are path-based [39, 56, 85]. These approaches

measure the data path to detect when the sending rate is too high, and then back-off the sending rate. In ICNs, measuring the data path is impractical. Because of replication-based forwarding and SCF, each message between the same sender and receiver potentially follows a different path. To use path-based techniques, a large combination of possible paths would have to be measured. On top of the difficulty in determining which paths to measure, any end-to-end feedback that the network tries to use will be highly delayed and unreliable making it difficult to rely on such information for accurate congestion control. Some existing SCF approaches statically limit the replication [121] in the network, but because conditions vary within the network itself, determining the best static limit is difficult. A new approach is needed that can adjust to changing network conditions without requiring end-to-end feedback.

Due to the lack of end-to-end paths and high-latency communication, congestion control is best implemented with local algorithms performed by individual nodes. Local, node-based congestion control for ICNs is designed in Chapter 3 that dynamically manages limited buffer storage space in replication-based SCF networks to improve network-wide message delivery/throughput. Using only local node-based decisions, network throughput can be improved by up to 280% in some scenarios, and 20% in nearly all scenarios.

2.2 Data Overlay

A new focus is emerging for networks in which data is the primary entity rather than devices. Instead of connecting devices to devices, as SCF does, a new service approach is emerging that concentrates on connecting users to data objects, irrespective of the underlying devices [76]. Such functionality has been provided in the world wide web using centralized databases and search engines where data objects are stored on servers that users can query directly. As end-users start to generate data from mobile devices such as embedded sensors, photos and updates like tweets, the upload bandwidth available to the UCN will be insufficient to handle all of the data. In addition, access to centralized services is limited by the intermittent connectivity. In situations where centralized infrastructure does not exist, distributed systems have been developed such as in sensor networks [92] and distributed overlays [64, 124, 112]. In these systems, data is stored at the end-nodes themselves. In order to determine where to store data and correspondingly find that data, structured overlay routing algorithms have been developed that repeatedly forward requests closer to the node where data is stored. However, while these approaches avoid the bandwidth bottleneck of centralized approaches, they too are vulnerable to intermittent connectivity.

The primary reason structured overlays fail in the UCN is that data objects are statically mapped to

specific nodes. This means to store and look-up data, the mapping must first be learned and then the data object or request forwarded to a single device. In many implementations, finding the mapping involves contacting several different nodes in the network. Given the high mobility rates in the network and the high churn rates of end-user devices, the storage node for a given data object will constantly be changing. In addition, the probabilistic nature of SCF message forwarding means control messages to maintain the overlay structure will be lost and in both cases high recovery overhead will be generated. Instead of deterministically mapping data to specific nodes, an opportunistic approach is needed that is immune to intermittent connectivity and device instability.

Chapter 4 presents Locus, a location-based data overlay for storing and accessing data within the UCN. Instead of attempting to maintain a structured mapping of data to nodes in the network, the overlay stores data in "bubbles of knowledge" around a home location for the data. Any nodes that are currently in the bubble of knowledge can hold the data. Data is always moving between nodes to try to reach the center of its bubble. Queries are made for specific locations instead of specific objects and forwarded using location-augmented SCF techniques. Because the data objects, queries and responses must all share the same channel, there is competition for the limited bandwidth during each node encounter. To overcome the limited contact duration and storage capacity of each node, new message policies are created that prioritize messages based on location of the node and data.

Chapter 3

Replication Management for Intra-network Message Forwarding

Message forwarding between mobile devices in the user-created network (UCN) is challenging because the device mobility leads to intermittent connectivity within the network. As a result, end-to-end paths between devices rarely exist making it difficult to provide reliable and high-throughput message delivery using traditional routing protocols. Store-carry-and-forward (SCF) routing increases the reliability of message delivery by treating disconnectivity as a natural part of the network. Multi-copy replication-based forwarding can help to increase the throughput of the network by increasing the delivery probability. However, determining the optimal rate of replication is difficult due to limited storage space on the mobile devices and the fluctuating network conditions within the UCN. In this chapter, a local node-based replication management protocol for SCF is developed that enables both increased reliability and message throughput for intra-network message forwarding while only using local information. The benefits of the approach are demonstrated in simulations of large-scale mobile networks.

3.1 Message Replication and The Buffer Constraint in Intermittently-connected Networks

The UCN is an intermittently-connected network (ICN) and is unstable and prone to partitioning and extended periods of disconnectivity [52, 110, 133]. To overcome these network constraints and support both delay- and disruption-tolerant applications, a new class of routing protocols for ICNs has emerged that forward messages using a store-carry-and-forward approach. Given the lack of stable end-to-end paths, message replication is commonly used to increase delivery [30, 37, 53, 93, 105, 121, 128]. However, the limited resources available to the mobile nodes can quickly be overwhelmed by too much replication which leads to congestion and ultimately to reduced delivery rates. Given the dynamic nature of such networks, dynamic replication management, in terms of when, which and how many messages to replicate at a given encounter, can be used to find a balance between over- and under-replication, both of which reduce the effectiveness of the network.

The goal of replication management is to maximize message delivery by avoiding congestion. However, the point at which congestion occurs is not static in an ICN because the network size, density and message generation rates often change frequently. Any static replication management will either over-replicate, causing excessive message drops and reducing message delivery when congestion is high, or under-replicate, missing potential delivery opportunities and under-delivering when congestion is low. Existing ICN protocols attempt to avoid congestion either by capping replication with a fixed quota [105, 121], intelligently filtering replication opportunities [30, 37, 53, 93] or, sometimes in conjunction with the other mechanisms, flushing already delivered messages from the network with acks [30, 37, 119]. While these approaches have proven to be effective in different scenarios, all of these replication management approaches are static and so cannot react to changes in network congestion.

The main challenge to effective replication management is the accurate detection of congestion in the ICN. In general, congestion can be defined as the point where network-wide delivery rate decreases due to an overload of network resources such as buffer space or bandwidth. In ICNs, buffer space is often overwhelmed by aggressive replication. While buffer management and flow-based feedback have been successful at preventing Internet congestion [56, 25], the lack of stable or even any end-to-end paths in ICNs eliminates the use of flow-based or end-to-end mechanisms. However, observations about global and local network conditions can be used to effectively reduce congestion and increase delivery rates.

An ideal ICN congestion detection scheme would monitor the entire network to learn the current congestion level and feed that information back to all nodes. However, because of high and variable message delays and unreliable delivery in ICNs, a global algorithm is impractical. Instead, a more attainable approach is to have each node act autonomously, using only *local* metrics to adapt its replication rate. A local congestion detection algorithm is still able to cope with spatial-diversity in congestion and can respond quickly to temporal congestion changes. The difficult part of this approach is to determine which local metrics can be used to capture the congestion state of the network.

The first step is to undertake a comprehensive study of the effect of diverse network conditions and limited buffer space on message delivery in ICNs. The model presented in this chapter enables the tracking of relevant global network metrics over diverse network scenarios, exposing which metrics indicate congestion in the network. Although not directly implementable as a replication management scheme in ICNs, this model leads directly to an understanding of which local metrics can be used by individual nodes to approximate the relevant global metrics, and so be used effectively in replication management algorithms. Given a clearer picture of network behavior, the second step is to design an algorithm for congestion control in ICNs, including congestion detection and replication management. The ICN congestion

		Buffer Size (KB)					
		1000	2500	3750	5000	7500	10,000
Message Period (sec.)	20	2	2	2	3	3	4
	40	2	3	4	5	6	9
	60	3	3	4	6	11	11
	80	2	5	7	9	13	18
	100	2	5	8	10	18	20
	120	3	6	11	16	20	20
	200	4	12	14	20	20	20
	300	9	18	18	20	20	20
	400	9	20	20	20	20	20

Table 3.1: The number of copies for Spray and Wait which gives the best delivery rate for the given message period and buffer size.

control algorithm in this chapter adjusts replication rates at individual nodes to maximize delivery rates. By allowing each node to react independently, the algorithm is responsive to both spatial and temporal fluctuations in congestion. Additionally, this congestion control is protocol-independent. It does not interfere with the forwarding decisions of the underlying routing protocol about which messages to send. Instead, it limits the number of messages a node is allowed to replicate during each encounter. Simulation over a diverse set of mobility scenarios is used to show that this approach is effective at improving network-wide delivery rates and does so with little overhead.

3.2 Congestion in ICNs

Congestion in ICNs is caused by over-replication in the network, which leads to buffer overflow at individual nodes and ultimately to reduced delivery rates in the network. While it is intuitive to approach the problem from a sender's perspective, allowing the sender to regulate replication for its messages, two problems arise. First, intermittent connectivity, path diversity, high loss rates and long response times make it difficult for the sender to react to congestion in a timely and effective manner. Second, replication is performed at every node that has a replica of the message. Therefore, a sender cannot know how every node that will receive a replica of its messages should react. These challenges make it ineffective to apply flow-based and end-to-end congestion control approaches from the Internet [25]. Instead, congestion control in ICNs becomes a problem of managing replication at every node in the network.

Effective replication management requires firmly limiting the number of replications allowed. Existing quota-based ICN protocols perform message-based replication management where each message sender gives a static initial quota for the maximum number of replicas of a message [105, 119, 121]. As messages are replicated, the quota is divided between the new and old replica until each replica has no quota left.

While using quotas certainly limits replication, the message sender does not know the congestion levels in other parts of the network or at later times when the message will be replicated. In dynamic networks, the best choice of initial quota can vary significantly depending on congestion conditions. For example, consider a simple network with 100 nodes, each using random waypoint mobility and routing with Spray and Wait [121]. By varying the maximum quota from 2 to 20 and varying the message rate and node buffer size from 1000KB to 10000KB in simulation, the initial quota that achieves the highest delivery rate varies dramatically, ranging from 2 to 20 as shown in Table 3.1. Compared to a fixed initial quota of 8, which offers a nice balance of low delay versus low overhead in a network of this size [121], the performance improvement by selecting the best quota value is on average 7% with a maximum improvement of 20%. Even in this simple network, it is clear that the quota that maximizes delay depends on the state of the network.

Given the limitations of message-based replication, a promising alternative is the use of dynamic node-based replication management in ICNs where individual nodes determine how many messages they can replicate at each encounter. The novelty of node-based replication management is that it is performed locally, independent of the routing protocol used. Essentially, a routing protocol orders the messages that it wants to transfer at each encounter and the congestion control algorithm determines how many of them are actually sent. To determine such local replication limits, nodes must observe the current level of congestion in the network. To answer the question as to which metrics are most effective to detect congestion in ICNs, a model of global network behavior is developed in the next section. Following that, a novel measurement-based local congestion control algorithm is presented and evaluated.

Although buffer management has proven effective for avoiding congestion in the Internet [56, 81, 88], those approaches are not applicable for controlling congestion in ICNs since they assume some interaction with end-to-end transport protocols. However, some existing ICN protocols do use buffer management in ICNs for dropping and forwarding according to a given policy [30, 37, 53, 93]. While policies can reduce the amount of replication in the network, in all of these protocols if every message in the buffer matches the policy, then every message will be replicated. Without knowing a firm limit of allowable replications, it will be difficult to avoid over-replication in congestion situations. Although not entirely orthogonal to the decision about replication limits, combining replication management with effective buffer management can dramatically improve delivery rates. The exploration of the behavior of different policies with congestion control is explored at the end of this chapter.

3.3 Global Congestion Behavior

Before the goal of developing a local congestion control algorithm can be achieved, it is necessary to first understand the behavior of the network under congestion. Given an understanding of the high-level global behavior, local approximations can be developed. The purpose of this study is to find metrics that can track the change in delivery rates due to buffer congestion. Given the large number of dynamic network metrics, the challenge lies in determining which network metrics are the most informative. In any type of network, a decrease in the number of successful or duplicate deliveries or acknowledged messages could indicate an increase in congestion. Similarly, a rise in message drops and buffer usage can also indicate an increase in congestion. In ICNs, the number of message replications or total message copies might also be indications of network congestion. The relation of these metrics to delivery rate depends on complicated interactions of different network parameters. Therefore, as an analytical tool to reason about global congestion behavior, a simple stochastic model is developed that simulates the delivery of messages in networks with constrained buffers. The model is an approximate representation of an ICN and provides interesting insights into the effects of congestion that inspire the design of the congestion control algorithm in Sec. 3.4.

An effective approach for modeling ICNs is to treat the spread of a message as the spread of a disease [48]. When two nodes have an encounter, they "infect" one another with new messages. While this modeling approach has been used before [118, 134], the models either do not account for the effects of buffer overload and message drops in the network, or are asymptotic and do not capture per-encounter behavior which is needed to track individual metrics. Other analytical models [30, 123] do not consider delivery rates and therefore cannot quantify the effects of congestion on successful delivery. In comparison, this model computes the network delivery rate based on the spread of messages throughout the network in response to changes in the input parameters network size, buffer size, contact rate, message generation rate and replication rate, each of which effect delivery and therefore are key to the model.

The model tracks message replicas using a Markov chain. Each state in the chain denotes the number of message replicas present in the network. Therefore, the Markov chain starts with potential states $\{0, ..., (N-1)\}$, where N is the number of nodes in the network. A special state `SUCCESS` represents the successful delivery of a message. When a message is first created it always enters the state 1. The final states of a message are either `SUCCESS`, for successful delivery, or 0, for no replicas remaining. State transitions occur as message replicas are copied, dropped or delivered. The probability of each state transition depends on the model input parameters and will be given in detail below.

Even after successful delivery, some message replicas will remain in the network. Therefore, the model

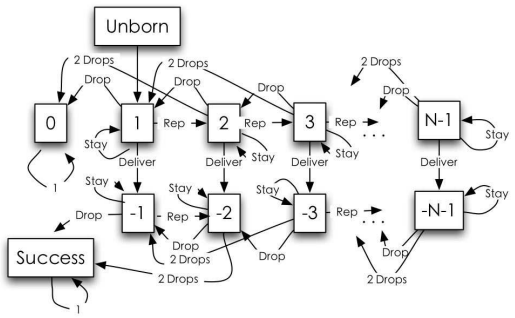


Figure 3.1: Markov chain representing the spread of a single message through the network. Each state represents the number of replicas of the message. Special states represent delivery of the message and the pre-creation state. State transitions occur on message drop, replication and delivery.

includes special states $\{-1, ..., -(N-1)\}$, where state $s \in \{-1, ..., -(N-1)\}$ indicates that there are $|s|$ replicas of the message remaining in the system after delivery. The final state -1 transitions into `SUCCESS` instead of 0 when the last message copy is dropped, since the message has already been delivered. For example, removing a replica of a delivered message transitions the state from $-s$ to $-(s-1)$. Alternatively, the model can be modified to allow for message flushing by leaving out the negative states. Note that upon successful delivery of a message, all copies of the message would be immediately flushed from all of the network buffers. Flushing delivered messages is a common practice [30, 37, 119], although this instantaneous model is overly aggressive and artificially lowers the congestion.

To accurately model drops the state of all messages must be known. The above model can be modified to include super-states which encapsulate the current states of every message. The probability to transition between a state is the probability that each message transitions into the new super-state. Messages are created sequentially during the simulation depending on the message generation period. The state `UNBORN` is added to signify that a message has not yet been created. Fig. 3.1 illustrates the complete Markov chain for a single message.

3.3.1 System Model

The network consists of N nodes and M messages created over time. Every node has the same size message buffer, capable of holding BS messages. Messages are all the same size. Because only the high-level network behavior is a concern, the data and mobility models are simplified to ease modeling. New messages are generated periodically with sources and destinations selected uniformly at random from the entire network. It is also assumed that encounters occur globally at a fixed period with each participating node selected uniformly at random from the network.

A random drop policy is implemented, so that when a node must drop a message, it selects one at ran-

dom to be removed. All nodes follow a basic Epidemic forwarding policy [128], where each node replicates to the other node messages it has that the other node does not have. In the model, messages are transferred instantaneously so that a message enqueued to be sent to the other node is not dropped by incoming messages before it can be sent. The effect of limited contact duration is not modeled, focusing instead on buffer space. Later simulations do include limited contact duration between nodes.

3.3.2 Transition Probabilities

The network is modeled as a discrete-time system with minimum time interval Δt. At every time step, the model transitions into a new state with some probability based on the current network state. Two network events can cause message transitions: a node contact or generation of a new message. To model these events, the contact and generation periods are first defined. The contact period is expressed as $C\Delta t$ and the message generation period as $G\Delta t$. It is assumed that only one event occurs during each interval. If two events do occur in the same interval, one is selected with probability 1/2. The probability of a contact event:

$$P_C = \left(\tfrac{1}{C} \cdot \left(1 - \tfrac{1}{G}\right)\right) + \left(\tfrac{1}{C} \cdot \tfrac{1}{G}\right)\left(\tfrac{1}{2}\right), \tag{3.1}$$

where the first term represents the probability that a contact event occurs with no generation event and the second term the probability that the contact event is selected when two events occur simultaneously. The probability of a generation event, P_G, is defined similarly.

The state of each message m_i is stored in s_i. When referring to a specific message or message state, the subscript m_i (s_i) is used. Otherwise, m and s are used to indicate a general message. State transitions are defined by the input parameters, BS, N, C and G. The initial state for every message is the `UNBORN` state, signifying that the message has not yet been created. A new message is created every time step with probability P_G. The probability of message m_i entering the network at a given time depends on the probability of message m_{i-1} having already been created. Explicitly, m_0 transitions from `UNBORN` to state 1 with probability P_G. For each successive message m_j, the creation probability is $(1 - Pr(s_{j-1} = \texttt{UNBORN}))*$ P_G.

Once created, a message can be delivered during any contact, which causes the state to transition from s into $-s$. The probability to deliver a message is:

$$P_{deliver} = P_C \cdot \tfrac{2}{N} \cdot \tfrac{s}{N-1}. \tag{3.2}$$

Essentially, a delivery happens during a contact event (P_C) in which one of the two nodes is the destination, and the other node has a replica of m.

The only other allowable transitions from state s are into $(s+1)$ upon replication, into $(s-1)$ or $(s-2)$ upon drop, or to remain in s. The transitions from $-s$ to states $-(s+1)$, $-(s-1)$ and $-(s-2)$ are the same. For contact events, a and b represent the two nodes involved in the contact event. Similarly, for generation events, a represents the node that generates the new message.

All of these transitions depend on the behavior of nodes during arrival of new messages. The following equations describe node behavior:

$$\alpha = \sum_{m_i} \left(\frac{s_i}{N}\right) \cdot \left(1 - \frac{s_i - 1}{N-1}\right) \tag{3.3}$$

$$P_{full} = \sum_{m_i} \left(\frac{s_i}{N}\right) \cdot \left(\frac{1}{BS}\right) \tag{3.4}$$

$$P_{drop} = P_{full} * \cdot \frac{R \cdot \alpha}{(BS + R \cdot \alpha)} \tag{3.5}$$

When two nodes encounter one another, each node has some number of messages that the other node does not have that might be transferred. α represents the expected number of messages to be transferred and is calculated as the sum over all messages of the probability that a has the message and b does not. After an encounter, a node will have to drop messages if its buffer is full. The probability that a node's buffer is full, Eq. 3.4, is the probability for each message that a node has the message, over the node's capacity. Finally, the probability of message m being dropped, Eq. 3.5, is the probability of it being one of the randomly chosen messages selected for removal when an overflow happens.

The number of messages actually transferred from one node to the other depends on the replication probability, R, which captures the probability that a particular message is chosen for replication. R ultimately determines the fraction of messages a node is allowed to replicate so that only $R \cdot \alpha$ messages are transferred during a given encounter. If the node's buffer is full, the same number of messages will have to be dropped. Therefore, m is dropped if it is selected among the $R \cdot \alpha$ removed messages. The probability that a node keeps m after a transfer is $P_{keep} = 1 - P_{drop}$. The parameter R in Eq. 3.5 allows nodes to control the replication probability and so manage congestion. In Epidemic, R is always fixed to 1. Later different values of R will be analyzed.

To transition from state s to $s+1$, there must be a contact event in which a replication occurs and both nodes keep m:

$$P_{+1} = P_C \cdot \frac{N-2}{N} \cdot \frac{((N-1)-s) \cdot s}{\binom{N-1}{2}} \cdot R \cdot (P_{keep})^2 \tag{3.6}$$

The contact must be between two nodes neither of which is the destination, which is expressed in the first two terms of Eq. 3.6. For a replication to happen, only one node can have m, the probability of which is the number of pairs with only one copy over all possible pairs, given in the third term. The last two terms say that the message must be selected for transfer, and then both nodes must keep their copy.

If, on the other hand, both nodes have m and both drop m, then s transitions to $s-2$:

$$P_{-2} = P_C \cdot \frac{N-2}{N} \cdot \frac{\binom{s}{2}}{\binom{N-1}{2}} \cdot (P_{drop})^2 \tag{3.7}$$

This transition only happens if the event is a contact event in which neither node is the destination. If both nodes have m, expressed as the number of pairs that both have m over all pairs (see Eq. 3.7), and both nodes drop their replica then s transitions to $(s-2)$.

Finally, the transition from s to $s-1$ is the most complicated transition. A single drop of m can occur during a message generation event, a contact event in which both nodes have m or a contact in which only one node has m:

$$\begin{aligned} P_{-1} = {} & P_G \cdot \frac{s}{N} \cdot P_{drop}(\alpha = 1) + \\ & P_C \cdot \frac{N-2}{N} \cdot \left(\frac{\binom{s}{2}}{\binom{N-1}{2}} \cdot (2 \cdot P_{drop} \cdot P_{keep}) + \right. \\ & \left. \frac{((N-1)-s) \cdot s}{\binom{N-1}{2}} \cdot \left((1-R) \cdot P_{drop} + R \cdot (P_{drop})^2 \right) \right) \end{aligned} \tag{3.8}$$

In a message generation event, a drop occurs if the generating node has m and m is selected for removal because of the new message. In a contact event between two non-destination nodes which both have m, a single drop occurs if either node keeps m and the other drops it. Finally, if only one node of a contact has m, it might replicate m with probability R. s transitions to $(s-1)$ if m is replicated and both nodes drop it, or if m is not replicated and the one node drops it.

3.3.3 Solving the System

To solve the model, the above transition probabilities between states are repeatedly applied to converge toward either the 0 or the `SUCCESS` state for each message. A transition matrix is created using the above equations. The current state of each message is represented by a vector of probabilities to be in each state. By repeatedly multiplying the transition matrix with the state vector, the probability distribution across states after a certain time can be found. When super-states are used to track all messages, there are $O(M^N)$ states which is unwieldy for meaningful networks. Instead, to solve the system a separate chain is main-

Algorithm 1 TRANSITION($M, states, endtime$)

Require: S = M state vectors of length *states*
Require: T = M transition matrices size (*states* x *states*)

```
1: for all messages m do
2:    S(m)(UNBORN) = 1
3: for t = 0 to t = endtime do
4:    for all messages m do
5:       T(m) = TransitionMatrix(m, S)
6:    for all messages m do
7:       S(m) = S(m) * T(m)
```

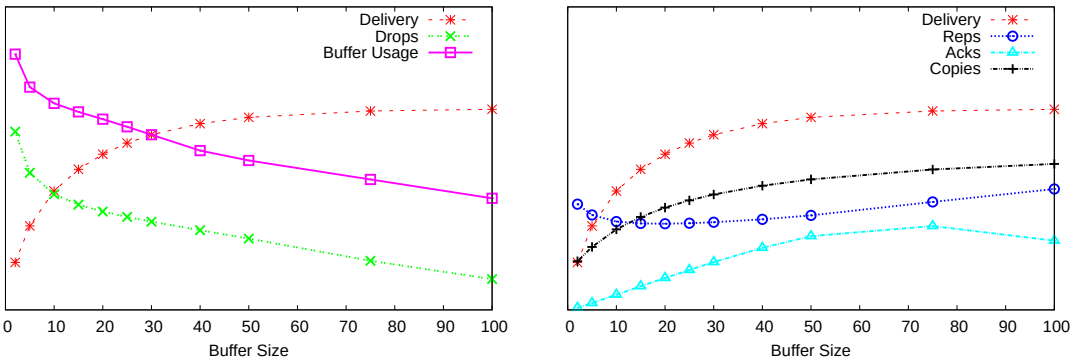
tained for each message and the transition matrix updated at every time step using the current states of the other messages. The current state of a message is computed as $\sum_{\text{state } t} Pr(s_i = t) * |t|$. Alg. 1 shows the process for solving the system.

3.3.4 Detecting Congestion

The goal of the model is to discover how different metrics change compared to the network-wide delivery rate and if they can be used to track congestion. At first glance it seems that an increase in network drops would be sufficient to indicate network congestion. However, consider the case when there is no replication at all. Drops will be very low, but so will delivery. As replication increases delivery will increase along with drops. This rise in drops is by itself not an indication of over-replication, i.e. congestion, since delivery is still increasing. Instead, the positive aspect of replication must also be consider in conjunction with the negative (drops). The model is used to evaluate different metrics as either positive or negative indicators of congestion.

To generate congestion the size of the message buffer is decreased. Because the model simulates individual node contacts, it is possible to count drops, replications, and max buffer consumption and track the spread of acks. The expected spread of acks at a given time t is the probability that m has been delivered times $a(t)$. The function $a(t)$ represents the copies of an ack after time t and is computed recursively as the probability that during a contact one of the nodes has the ack and the other does not. The probability of having the ack depends on $a(t-1)$ where $a(0) = 1$.

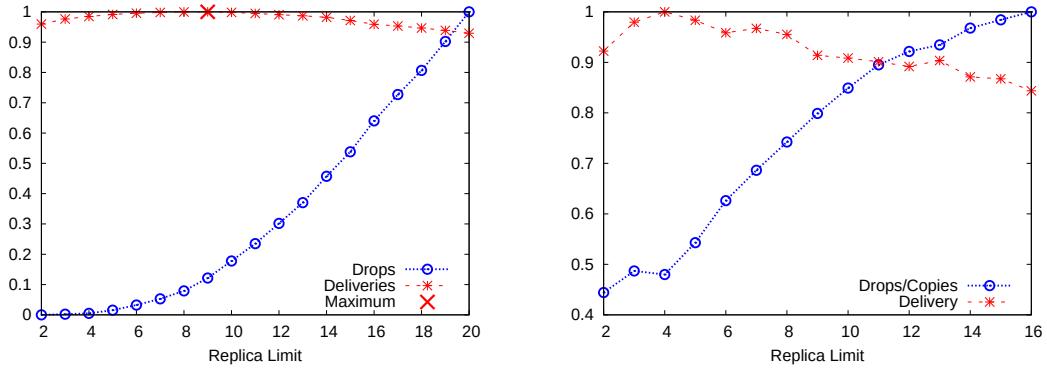
To discover the behavior of these metrics, the model was solved using Alg. 1, as shown in Fig. 3.2. The metrics are separated in the Figure with respect to how they change in response to congestion. Drops and buffer use show continuous increase with congestion, while acks, message copies and replications tend to decrease with congestion. At first glance, it would seem that using one of the metrics from Figure 3.2(a) would suffice to track congestion in the network. However, these metrics only reveal the negative impact of replication. In some cases, the positive impact from replication outweighs the negative impact, as shown



(a) Metrics Increasing with Congestion

(b) Metrics Decreasing with Congestion

Figure 3.2: Change in metrics as congestion is increased by varying buffer size. Metrics that indicate an increase in congestion (a) are shown separately from metrics that indicate congestion decrease (b).



(a) Drops as Congestion Measure

(b) Drops/Copies as Congestion Measure

Figure 3.3: Normalized results of using different combination of metrics to track congestion. Drops only (a) misses the positive impact of replication. Using both drops and message copies (b) is more accurate.

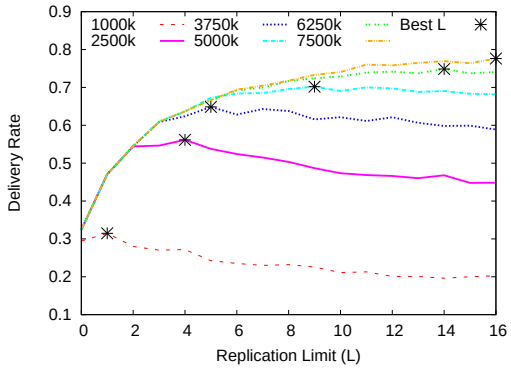


Figure 3.4: The change in delivery rate as L is varied with different buffer sizes. The best L value is marked for each scenario.

in Figure 3.3(a). As the replication rate increases, the delivery rate increases. However, drops is also always increasing. Congestion does not occur until replication limit reaches 9, but based on drops alone, the increase in drops starts at 4. Instead, the negative impact of replication must be balanced with some measure of the positive impact. Therefore, metrics from both Figure 3.2(a) and Figure 3.2(b) must be combined. The ratio of drops/copies in relationship to delivery is shown in Figure 3.3(b). Two observations from outside the model helped determine which specific metrics to use from the possible set. In most simulations, the buffer consumption is always near 100%, even in low congestion. Also, the spread of acks is unreliable and delayed and the behavior of replications does not align with delivery rates. Therefore, the ratio of drops over copies is used to track congestion.

3.3.5 Limiting Replication

The discussion of the variable R remains until now. R represents the probability of forwarding a message and is used to limit replication. R is the only parameter that nodes can change and therefore the means by which replication management is done. The function for R depends on the replication management approach. The node-based limiting approach specifies the maximum number of messages, L, transferred during each contact. The probability of m being forwarded is the probability that m is one of the L messages selected for transfer. Assuming a random forwarding policy, $R_{limit} = \min\left(\frac{L}{\alpha}, 1.0\right)$, where α is from Eq. 3.3.

By varying L and computing the delivery probability in different scenarios, it was found that the value of L resulting in the best delivery ranged from 1 to the maximum value. However, because of the simplifications of the model, the full performance impact is not captured. To fully understand the possible impact of node-based replication management via L, simulation is used. By fixing the replication limit to different values the delivery rates at different levels of congestion can be compared. Choosing the wrong value of L

can result in a 10-40% change in delivery rates (see Fig. 3.4).

3.4 ICN Congestion Control

Given the disconnected and dynamic nature of ICNs, congestion control must act locally while thinking globally. Our model in Sec. 3.3 highlights which global metrics can track congestion. In this section, we present our local congestion control algorithm based on those observations. Each node independently calculates a local approximation of the current congestion level, CV. As the node samples the network, it continually updates CV. Whenever CV is updated, the node also adjusts the replication limit L following an additive increase, multiplicative decrease (AIMD) algorithm.

To maintain CV, local metrics are needed to approximate the global metrics for drops and replications. Global drops can easily be approximated by measuring the local drops at an individual node. On the other hand, approximating global replications is not as straight forward. When a node replicates a message the node does not know for certain that the other node kept the replica. Therefore, replications are counted by successful *incoming* replications. The fidelity of these two metrics can be improved by including measurements from other nodes. When two nodes meet they exchange their current drop and replication counters. In addition, because a message is replicated along each hop it travels, the hop counts of stored messages give further replication information.

To calculate the congestion value, a node monitors the network for a certain time, and then computes CV' which is the ratio of drops and replications collected in the last sample. The drop and replication counts are reset for each sample period. To dampen the effects of temporary spikes in congestion, the new congestion value is calculated using an estimated weighted moving average (EWMA) with $\alpha = .9$ so that $CV' = \alpha \cdot CV_{sample} + (1 - \alpha) \cdot CV$. The most recent sample is given higher weight to keep CV fresh.

After calculating CV', the node adjusts the replication limit. As CV moves up and down, the replication limit should grow to take advantage of all available resources, but back-off when congestion increases, similar to how TCP updates its congestion window [25]. Because changes in the replication limit propagate slowly through the network, the limit should grow gradually so as not to overwhelm the network. If the new congestion value CV' is higher than the previous value CV, there is growing congestion and the limit is reduced by multiplying the current CV by multiplicative factor ($md < 1.0$). If, instead, CV' is less than CV, the limit is increased by a fixed amount $ai > 0$. Experimentally, it was found that a lower md value, i.e. stronger back-off, coupled with a low ai, i.e. slower growth, resulted in the highest delivery rates in congested scenarios. We use $md = 0.2$ and $ai = 1$.

Algorithm 2 ProcessEvent(*event*)

Require: $drops = 0$
Require: $reps = 0$
Require: $limit = 1$
Require: $CV = \infty$
Require: $ai > 0, 0 < md < 1.0$

```
1: if event = Drop then
2:    drops = drops + 1
3: else if event = Receive Message then
4:    reps = reps + 1
5: else if event = Contact with node b then
6:    d = drops + b.drops
7:    r = reps + b.reps + Σ_{m∈stored messages m} (hops(m) − 1)
8:    reset(drops, reps)
9:    CV' = α · (d/r) + (1 − α) · CV
10:   if CV' ≤ CV then
11:      limit = limit + ai
12:   else
13:      limit = limit ∗ md
14:   CV = CV'
```

The choice of time length to sample the network is important to adjust the replication limit at an appropriate pace. CV should be updated as frequently as possible, but not too quickly to lose meaning. The minimum time to detect change in congestion is the minimum time for the congestion metrics to change. As illustrated in Sec. 3.3, replications and drops can only occur during node contact and message generation. Since no replication occurs in message generation, the minimum time to detect change in both metrics is one contact. Whenever a node contact occurs the nodes update CV. Pseudo-code for the algorithm is shown in Alg. 2.

3.5 Simulation Results

The goal of the evaluation is to show the delivery improvement attainable using congestion control across various degrees of congestion and in diverse network configurations. The effectiveness of node-based congestion control was evaluated by integrating it with several different routing algorithms. Each protocol was run with and without congestion control to demonstrate the protocol independent property of the congestion control algorithm. Epidemic [128] was the baseline protocol with no congestion control. For policy-based congestion control, the Prophet protocol was used, which selects messages for forwarding based on the likelihood of the contact delivering the message [93]. For message-based congestion control, Spray and Wait was selected which assigns a static initial quota to each message and distributes the quota in half at each encounter [121].

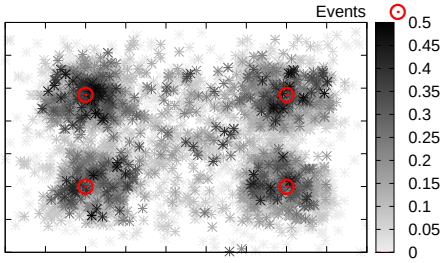


Figure 3.5: Map of calculated CV at specific points in the "Events" scenario. Higher congestion values, indicated by darker points, are mostly clustered around the four event locations.

3.5.1 Experimental Set-up

To perform the evaluations, a transport layer was added in the ONE simulator [82]. The transport layer is responsible for performing all of the congestion related functions, assigning sequence numbers to messages and managing message acks. Acks are used to flush already delivered message replicas from the network. The transport layer is queried whenever the routing layer wants to send a message. If the replication limit has not yet been reached for the current contact, the transport layer allows the transmission. When a message is delivered to the final destination, a new ack is generated and inserted into a fixed size cache. Each node stores all received acks in its cache along with the drop and replication counts. At the start of a contact, the transport layer sends the cache to the other node, which merges the information with its own. Any messages for which an ack has been received are removed. Then the replication limit is updated for the ensuing contact.

In all simulations, new messages were generated following a Poisson process at each node with mean message period varying based on the simulation. Each message was 50KB and the destination was selected at random from the network. Each node had a buffer of 1MB. To vary the level of congestion in the network the mean per-node message period was changed. Each simulation had 100 nodes in a 4500 x 3400m area with transmission range of 150m and speed of 250kbps.

To capture various network conditions, several scenarios were used to determine nodes' movement. In the baseline case, random waypoint mobility was used with speeds between 2 and 15m/s The other scenarios create spatial or temporal variations in congestion. The first scenario contains several events in different locations. Nodes congregate around the event locations, increasing congestion due to higher neighborhood message generation rate. When nodes are near events they also increase their generation rates, so that a node's message generation rate is a function of the distance to the nearest event. This scenario,

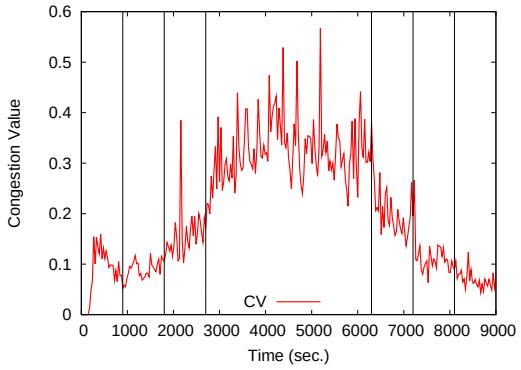


Figure 3.6: Change in CV over time as the message generation rate fluctuates in the "Diurnal" scenario.

which is called "Events", loosely models the behavior of mobile agents in a disaster [106]. In the second scenario, the message generation rate is changed at regular intervals to temporally vary congestion. For the first steps, the rate is doubled. Once the peak rate is reached, the network remains in high traffic load for several steps before returning to the original low rate by halving at each step. This behavior simulates a day-time/night-time traffic pattern. This scenario is called the "Diurnal" scenario. Nodes move following random waypoint movement.

3.5.2 Accuracy of Congestion Detection

Before evaluating the effectiveness of the full congestion control algorithm, first the local congestion detection component is validated in isolation. Congestion control was run with CV calculation, but without the replication limit. Whenever a node updated CV, the timestamp, location of the node and CV were recorded.

In the "Events" scenario, congestion should be higher near areas of high node density, for example, at the event locations. In surrounding areas the congestion should decrease. The congestion detection algorithm was able to capture this behavior, as can be seen by the map in Fig. 3.5. Each point is CV at that location in the space. Darker points indicate a higher CV. Interestingly, although the majority of high congestion was around the event locations, there were other areas of congestion in the spaces between events. This underscores the importance of dynamic, node-based congestion control because of unpredictable congestion conditions.

In the "Diurnal" scenario, node densities are fairly uniform due to the random waypoint movement. Changes in congestion instead come from increasing and decreasing message generation rates. The congestion detection algorithm successfully captures both the increase and the decrease in congestion, as seen in Fig. 3.6. The mean CV values across the network were computed every 30 seconds. At the beginning

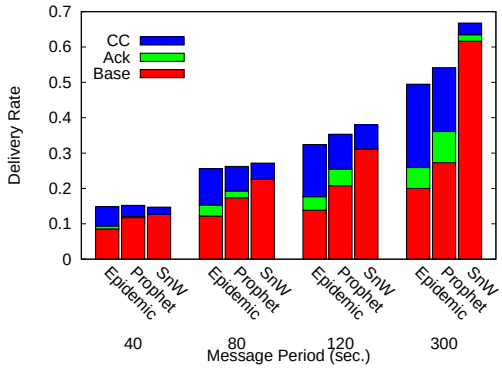


Figure 3.7: Breakdown of delivery rate in varying congestion from each component: the base protocol (Base), acks (Ack) and congestion control (CC).

there is a learning period as nodes begin to sample the network. After the initial period CV responds fairly quickly, moving both up and down in only a few update periods.

3.5.3 Benefits of Congestion Control

Next the full congestion control algorithm was evaluated including dynamic replication limiting. First the baseline scenario with uniform congestion was considered. Some benefit of the algorithm comes from freeing buffer space by using acks to flush unnecessary messages. To quantify the benefit from the ack and congestion control components, each protocol was run with full congestion control, with only acks and by itself. By reducing the message period, congestion was increased.

The experiment revealed several things (Fig. 3.7). First, at very high congestion, acks do not improve delivery because they do not propagate before the replicas have been dropped (see the Base+Ack bars). As congestion lessens, there is more benefit from acks, but the majority of delivery improvement comes from congestion control. Second, dynamic node-based congestion control improves delivery for each protocol in every scenario. Also, the effectiveness of node-based congestion control compared to policy- or message-based can be seen in the improvement of Epidemic. Epidemic by itself is very inefficient, but with node-based congestion control (Epidemic+CC) it generally outperforms the base policy-based (Base-Prophet) and message-based (Base-SnW) protocols in nearly every scenario. Despite the good performance of the basic Spray and Wait protocol, dynamic node-based congestion control improves delivery in all cases by at least 8%. Third, at the highest level of message generation the network is overloaded as seen by the low delivery rate. Newly generated messages quickly push stored messages out before the old messages have time to spread through the network. Even though the network can hardly operate, congestion control improves delivery by 15-73%.

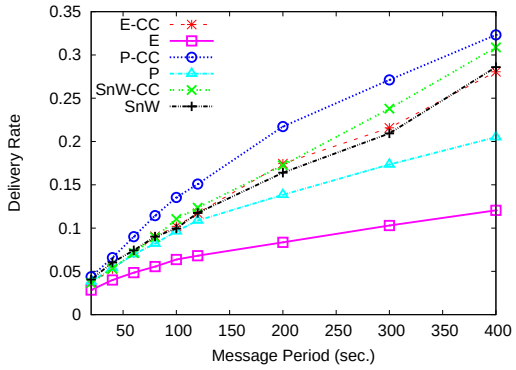


Figure 3.8: The change in message delivery rate in the "Events" scenario as the message period increases with congestion control (CC) and without.

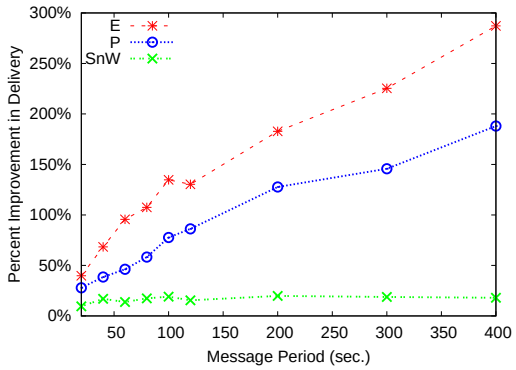


Figure 3.9: The percentage improvement using congestion control over the base protocol as base message period increases in the "Diurnal" scenario.

Examining the performance of the different protocols under spatial congestion shows the benefits of node-based congestion control even more. In the "Events" scenario, node-based congestion control is well-suited for the changing congestion. All protocols with congestion control perform better or as well as all base protocols (see Fig. 3.8). Node-based congestion control does especially well when coupled with policy-based forwarding. This hints at the benefits of choosing an effective forwarding policy to pair with congestion control.

In the "Diurnal" scenario, the relative performance of the protocols is very similar to the "Events" scenario. Because of dynamic replication management, congestion control is always able to adapt to the current message load. This is illustrated by the percentage improvement when using congestion control compared to the base protocols, as shown in Fig. 3.9. In low congestion levels, dynamic congestion control is able to take advantage of the extra available resources, increasing Epidemic by up to 280% and Prophet by 190%.

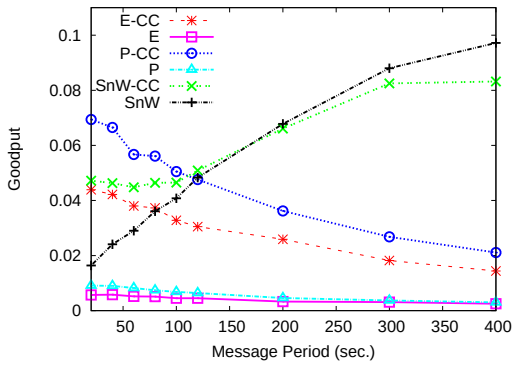


Figure 3.10: Goodput of each protocol with and without congestion control (CC).

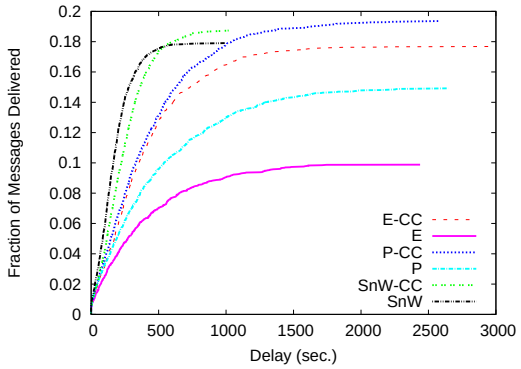
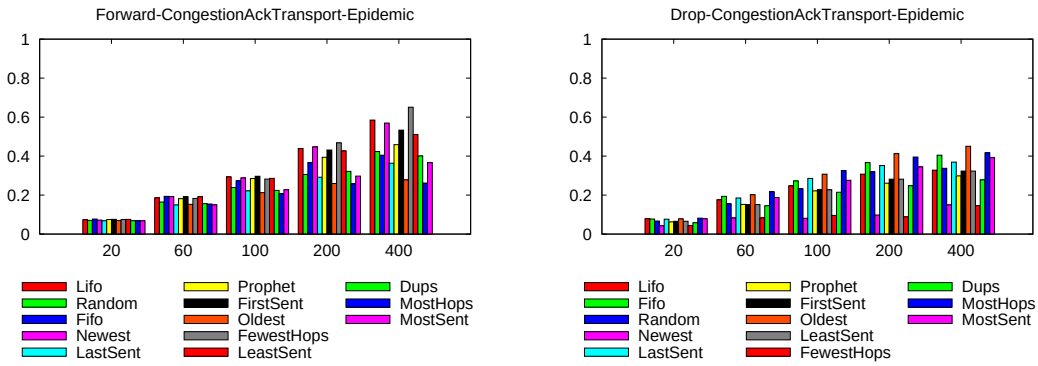


Figure 3.11: CDF of message delays for delivered messages with different protocols with congestion control (-CC) and without.

3.5.4 Overhead

Finally, to evaluate the impact that congestion control has on the underlying protocols, change in goodput and delivery delay is compared when using congestion control. Congestion control adds overhead from acks. In addition, because replicas remain in the system longer, each message might be copied more often. The goodput, bytes delivered over total bytes sent, quantifies the trade-off between overhead and improved delivery. When operating in high congestion scenarios, congestion control limits the replication, thereby lowering overhead. This leads to a higher delivery rate and a higher goodput (see Fig. 3.10). As the congestion lowers, congestion control increases the replication limit, pushing down goodput. Used with the flooding protocols congestion control always has better goodput. Spray and Wait, because of its message quota, has a relatively fixed overhead and so delivery rate grows faster than overhead. However, in high congestion congestion control has a significant impact on the overhead of Spray and Wait.

By limiting replications, congestion control increases the message delivery delay, but messages that otherwise would not be delivered are. This behavior can clearly be seen by examining the CDF of message



(a) Forwarding Policies With Epidemic Limiting

(b) Drop Policies With Epidemic Limiting

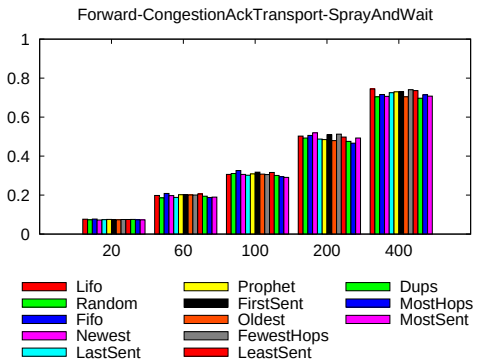
Figure 3.12: Performance of different forwarding (a) and drop (b) policies in conjunction with congestion control and epidemic (no) limiting.

delays between protocols from one run of the baseline scenario (see Fig. 3.11). The increase in delay is especially noticeable compared to Spray and Wait which is designed to reduce delay. However, the median increase in delay using congestion control compared to base Spray and Wait ranged at worst from only 6% to less that .1%. Because of some far outliers, the mean ranged from 237% in the worst case, to actually reducing delay by 10%. For Epidemic, the median increase in delay ranged from 3% to less than .1% and for Prophet the median increase was never more than .1%.

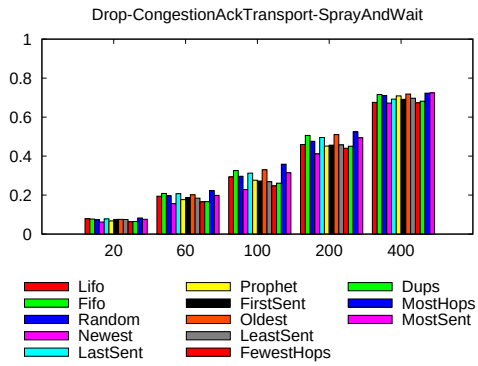
3.5.5 DTN Policies

Congestion control is successful in improving network delivery rates because it reduces the pressure on limited buffers by instead limiting contact opportunities. The result is that scenarios in which there was only one limited resource (buffer) now have two limited resources (buffer and contact duration). Although congestion control balances the strain on the two resources, it does not eliminate any of the constraints. Therefore, how those resources are used is very important. There are three policies which affect the consumption of limited resources in DTNs. Limiting policies, which determine the set of messages available for forwarding during each encounter, manage the energy resource by reducing the total number of bytes (messages) sent. Forwarding policies manage limited contact duration by sorting messages to allow high utility messages higher priority. Finally, dropping policies manage the limited buffer by removing messages with the lowest impact on delivery rate. Many policies have been proposed for each of the three categories. In fact, any existing DTN forwarding protocol can be broken down into a combination of these three policies. In this section a comparison of different policies and their effects on network delivery rates is executed.

Many such policies have been proposed in the literature [30, 37, 53, 71, 93] and, as demonstrated in

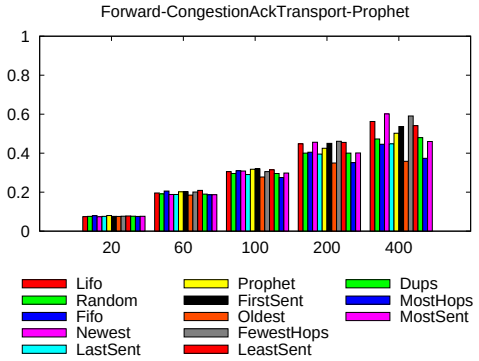


(a) Forwarding Policies With Spray and Wait Limiting

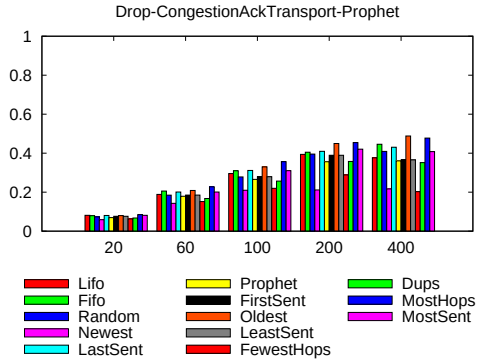


(b) Drop Policies With Spray and Wait Limiting

Figure 3.13: Performance of different forwarding (a) and drop (b) policies in conjunction with congestion control and Spray And Wait limiting.



(a) Forwarding Policies With Prophet Limiting



(b) Drop Policies With Prophet Limiting

Figure 3.14: Performance of different forwarding (a) and drop (b) policies in conjunction with congestion control and Prophet limiting.

Section 3.5, combining the correct policies with congestion control can lead to even higher delivery rates. The improvement in delivery comes from intelligently managing drops and the limited number of messages can be forwarded at each encounter. The order in which messages are forwarded has a big impact on overall message delivery, as does the selection of messages for dropping. To determine how these different protocols are affected by congestion control, and which policies can lead to the best delivery rates, the individual policies were studied in the existing framework.

Each message selection policy was used as both a forwarding and dropping policy. The policies were then combined with one of three limiting policy, based on existing ICN protocols, Epidemic which is no limiting, Spray and Wait which is quota-based limiting and Prophet which is limiting based on contact probabilities. The results are surprising, showing that simple policies are the most effective (see Figures 3.12, 3.13 and 3.14). In nearly every scenario, forwarding the newest message leads to higher delivery rates. Similarly, deleting the oldest message increases delivery rates even over more complicated schemes like

Prophet or even approaches that attempt to track the message copies in the network (LeastSent, FewestHops, etc.).

3.6 Future Work

One challenge in the current implementation is determining the correct multiplicative back-off value, *md*. When congestion is high, it is best to back-off quickly. However, at a certain point the network quickly switches to an uncongested state and the strong back-off in replication is too aggressive and actually hurts delivery. As a future extension to the existing congestion control algorithm, methods to dynamically scale *md* to prevent over- or under-reacting to congestion should be investigated.

3.7 Conclusion

There is a clear opportunity to improve the delivery rate of SCF protocols operating in ICN networks by responding directly to congestion in the network. In this chapter, it has been shown that congestion in an ICN can be detected by tracking the ratio of drops over replications. Using this mechanism, a novel, dynamic node-based congestion control was developed and shown to increase delivery rates (throughput/reliability) by up to 280% in some scenarios, out-performing all existing congestion avoidance schemes.

Chapter 4

Location-based Intra-network Data Overlay

Wi-Fi-enabled and sensor-equipped mobile devices allow new opportunities for participatory sensing in which end-user devices are used to gather real-time information about the environment and collaboratively share the data with other users for applications such as real-time vehicular traffic planning, environment monitoring and location-based announcements and advertisements. Providing high-reliability access to the large volume of collected data is difficult because of the limited wide-area bandwidth that prevents centralized approaches from being used. Given the collaborative nature of user-created networks (UCNs), a decentralized data overlay on top of the mobile devices is a natural solution. However, intermittent connectivity between nodes makes deploying existing data overlays inefficient and unreliable. In addition, end-user devices have limited storage and connectivity opportunities that can be quickly overwhelmed by the data overlay. In this chapter, a location-based data overlay for UCNs is developed that achieves high reliability and look-up throughput using location defined message utilities to maximize the local storage and connectivity.

4.1 Data Overlays in Intermittently-connected Networks

The wide deployment of embedded sensors and participatory sensing by end-users is enabling the generation of unprecedented amounts of data. User demand for access to these large volumes of data is creating a shift in focus towards content- and data-centric systems and away from traditional end-to-end communication [77]. In these new environments, the primary entities of the network will be data objects rather than individual devices. With this shift in focus and the low cost and small form factor of current sensors comes new opportunities for data collection and dissemination. A near-term future can be envisioned in which most vehicles and smart phones are equipped with an array of various sensors collecting data about the surrounding environment. Given the participatory nature of such environmental sensing, data is inherently tied to the location where it was generated and not to the user who created it. User-created networks are a natural mechanism for storing and sharing this data since they are built on similar collab-

orative mechanisms. However, in a mobile environment with limited resources, the accumulation of such location-specific data raises two important questions of where to store the data and how to access it.

At first glance, remote access, where data lives on centralized servers (e.g., Cartel [72]), seems to be a natural choice. Using a mobile connection such as 3G or WiMax, the collected information can be uploaded to central storage and retrieved by other nodes from the same storage. However, with the increasing number of devices generating data and the high rates at which data is generated, the bandwidth requirements will quickly overwhelm the infrastructure, especially considering that those networks are already pushed to the limit to serve existing mobile Internet access.

Instead, given the location-centric nature of the data, a better alternative approach is to store data on the mobile devices themselves. This has the benefit of reducing load on the congested wide-area infrastructure and also keeping the data near the devices that generate and consume it. Many distributed data overlays have been developed for the Internet that store data on end-nodes (e.g. Chord [124] and Pastry [112]). However, current data overlays make restrictive assumptions about node connectivity. The intermittent connectivity in networks of mobile devices [52, 110, 133] renders most of these solutions ineffective. Essentially, current overlay solutions are inherently node-focused, mapping data objects to specific nodes in the network. In vehicular networks where nodes have high speeds of mobility and quickly move through pockets of connectivity and other intermittent-connected networks, end-to-end communication between two specific nodes is difficult. What is needed is a data overlay that can run on top of an intermittently-connected network (ICN).

The location-based nature of the collected data, where data is inherently tied to a specific location or region, points towards caching of data in the geographic area where it was generated. Rather than moving objects to particular nodes in the network, the data objects are stored on whatever devices are available in the data's home geographic area. The goal is to keep each object as close to its home location as possible, given the currently available storage options. To find data, a node simply sends a request to the target location instead of searching for a particular data or node id. This approach tightly couples data to the location it describes, enabling a new class of location-specific applications such as real-time route planning, live monitoring of carbon footprint and infrastructure or location-targeted advertisement and data sharing. The challenge to enabling a location-based data overlay comes from the high mobility in the network. Since devices are always on the move, the set of nodes near an object's home location is in constant fluctuation.

This chapter presents Locus, a location-centric data overlay for intermittently-connected networks that stores data at physical locations and provides data look-up through location-based forwarding. The main

novelty of this approach comes from decoupling the data from the nodes that carry it. Data in Locus is explicitly tied to the location where it was collected. As devices leave and enter different areas, they exchange data with passing nodes using the location information available to them to prioritize and filter the set of messages that they will copy to the other node. Essentially, nodes pass stored data to other nodes that are in a better position for that data and receive data that lives near their future location. Support for this dynamic caching is enabled using replication-based techniques similar to message forwarding protocols in ICNs [30, 78, 93, 121, 122]. However, replication has been adapted to support data storage and location-based query/response rather than generalized end-to-end connectivity.

Given the limited contact durations between nodes in ICNs, the real challenge comes from the fact that the communication channel is shared between data object replication for storage and query and response message forwarding for look-up. Locus strives to balance the two demands to preserve object lifetime yet allow sufficient progress for message forwarding. To achieve this, Locus uses a utility-based message prioritization scheme for forwarding and dropping data objects based on the data's home location and the node's current location. Combined with these prioritization functions, alternate utility functions for query and response forwarding serve to strike a good balance between extending data availability and increasing query match rate and response delivery.

Simulation results in networks of moving vehicles show that Locus's policies are effective at keeping data near its home location. In turn, Locus's storage techniques, when coupled with location-based forwarding, lead to high query match rates and response delivery rates, achieving nearly 4 times higher query success rates compared to regular SCF forwarding approaches.

4.2 Data Management in ICNs

Location-specific data is the enabling component of a new class of applications. Access to near real-time and recent historical data about a specific location can be used to drive applications such as optimized traffic route planning, real-time fuel efficiency and location-based data sharing such as advertisements, announcements and media sharing. All of these applications need fine-grained information about specific locations for the best operation. In addition, these applications are dependent on up-to-date information, since parameters such as traffic and weather conditions can fluctuate quickly and have a large impact on application decisions. What is needed is an efficient system that can collect, store and publish the large volumes of data that will be generated and used by such applications.

4.2.1 Centralized Approaches

Many existing systems use a centralized or infrastructure-heavy approach to store data. One class of approaches assumes that all nodes are one hop away from a relay that can provide access to servers on the Internet [72, 135]. In addition, multi-hop solutions have also been proposed for communication between the vehicle and the relay [57, 72, 74, 129, 136]. However, all of these solutions require extensive deployment of new base stations and infrastructure that will require a high deployment cost or assume that data traffic can be carried by cellular networks.

The reality is that the load generated from these networks will be too much for existing cellular deployments. For example, in Chicago with an average demographic density of 5000 people per square kilometer, there are on average 2500 vehicles per km^2. If only 10% of those vehicles are generating data, the network must be able to support 250 data streams per km^2. A typical basestation coverage radius is several hundred meters, or approximately 100 users. Given an upload capacity of around 500Kbps per basestation, that leaves only 5Kbps per device. While this data rate can support simple sensor readings, transferring larger data objects like images or mp3s will exceed the capacity of the network. Although emerging 4G networks have higher upload capacity, the current deployments cover a larger physical area and therefore divide the capacity across more users. More importantly, whatever capacity the basestations do have is also shared with competing applications for mobile Internet access. Given that the demand for mobile Internet access will only increase, cellular-based approaches for location-based applications likely will not have sufficient bandwidth.

4.2.2 Decentralized Approaches in Connected Networks

Instead, decentralized, in-network data overlays can store data and keep it off the already burdened infrastructure while still retaining high data availability and accessibility. There are two requirements to provide an in-network data storage system. First, effective storage decisions are needed to ensure the availability of data. Second, effective query and response mechanisms must be developed to allow users of the system to find the data after it has been stored.

In-network storage has been done before in distributed databases for sensor networks. The focus in these systems has been on replica placement to improve data lifetime, reduce look-up delay and reduce network overhead [67, 116, 126, 132]. As storage solutions, these replica placement strategies are focused on overcoming energy constraints and high device failure and are not well-suited to handling limited storage and connectivity. Data access in these systems is typically assumed to be single hop or sink-based, where each sensor has a data path to the data collection sink. In the Internet, distributed hashtables (DHTs)

provide both storage and look-up functionality using structured routing techniques based on object and device ids [64, 112, 124]. The structured routing decides which node is responsible for storing a data object and ensures that queries for the object are forwarded to the correct node.

However, in an intermittently-connected network like those formed by mobile devices, connectivity is difficult and unreliable. The changing density of the network, node mobility and limited connectivity between devices render end-to-end approaches that assume stable connectivity ineffective. In addition, any system running in an ICN must cope with limited storage capacity, disrupted connectivity and limited bandwidth from short contact durations. Given that both DHT routing and existing sensor network systems require connectivity between nodes, the probability of finding data using these techniques in ICNs is very low.

4.2.3 Disconnected Networks

What is needed are data management and message forwarding approaches that can overcome limited connectivity, bandwidth and storage. Previous ICN research has addressed these issues, but in a different context. Because existing work has focused on end-to-end communication, the data management techniques are often destination-oriented [30, 37, 53, 93, 105] and not useful for performing data storage. While some replication schemes use only message characteristics such as message age or number of copies made [53], it is unclear how to effectively control the replication for data storage.

Even when performing data look-up in the ICN overlay, the destination-based focus of existing ICN protocols makes them ill-suited to use. To directly apply destination-based forwarding, a look-up service must be implemented that tells which node holds which data object. Maintaining and accessing such a service will be unreliable due to the lack of stable end-to-end connectivity. Without knowledge about the data object location or network state, single-copy routing [46, 49, 78, 80, 130, 136] will be unreliable. Multi-copy forwarding has been developed to operate without high levels of network knowledge [30, 37, 53, 93, 105, 121, 122, 128], but it is designed to probabilistically find devices instead of data objects. Unless there is a synergy between the data storage and data look-up, the probability to find data objects will be low.

The real problem with existing solutions is that data is mapped to nodes in a manner that is not primarily related to the nodes' proximity to each other. As nodes move, disrupted connectivity makes it difficult to find a particular node. Because routing between nodes and node availability in ICNs is so unstable, a new data mapping is required. Locus takes a simple approach that does not require information about the network and can handle the limited resources of ICNs. Locus stores a data object on the nodes closest

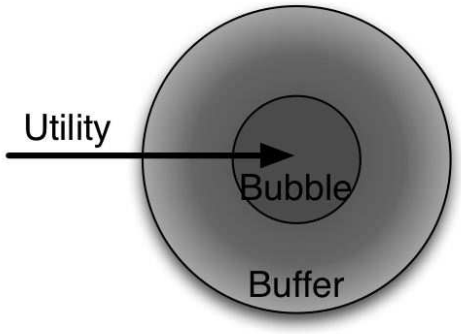


Figure 4.1: Storage bubble for a data object. Utility is highest at the center of the bubble. Utility drops off following a gradient around the edge of the bubble, creating a buffer that pushes objects back into their bubble.

to the object's home location. When two nodes meet, they replicate data objects in an attempt to keep the objects as close to the home location as possible. Look-up requests for a particular data object are sent to the location where data originates. In this way, end-to-end connectivity with specific devices is not needed.

4.3 Locus Data Storage

The primary entities in Locus are data objects. As mobile devices move throughout the environment they periodically create new data objects. When data is created, it is tagged with the current timestamp as well as the current location, which becomes the *home location* for the newly created object. An individual data element d is defined to be the tuple (χ, τ, ζ), where χ is the home location of d, τ is the time d was created, and ζ is the value of the data (for instance, the temperature reading if the data is a sensed temperature data point). It is assumed that each device is equipped with a GPS sensor to capture the device's current position. Although most new automobiles and mobile devices are equipped with GPS receivers, other location detection mechanisms could be used [44]. Although Locus can operate with any location measuring approach, applications benefit from more accurate measurements that allow fine-grained information to be collected.

Given a data object d, the main functionality of the data layer of Locus is to maintain the storage of d at or near the location(s) where it is needed. A significant innovation therefore lies in implementing virtual location-centric storage of sensed data. This virtual storage is managed through intelligent (i) replication management policies, which determine what data is forwarded to a node during a given contact opportunity, and (ii) cache replacement policies, which determine what data is dropped.

Instead of storing each object on a particular node, Locus copies the object to whatever nodes are cur-

rently near the object's home location. Due to node mobility the set of nodes near the home location is always changing. Because of this, Locus attempts to keep all objects within a "bubble of knowledge" around their home location. Locus replicates data objects to multiple nodes in or near the object's bubble to ensure availability. The goal of replication is to maintain the objects as close to their bubbles as possible.

4.3.1 Data Utility

Since data objects should be stored in the nodes near the object's geographic home location, the probability of a node maintaining the data diminishes the further the node travels from the home location for the data. To enable this smart data caching, utility functions are used that are associated with data based on the time and location where the data was generated. The relation of an object's utility to its bubble is shown in Figure 4.1. Objects closer to their home location have higher utility. The size of the bubble, γ, determines how far away from the home location an object can be stored. As the distance from the home location increases, the utility falls until a certain cut-off point where utility is 0. Around the bubble there is a gradient of utility creating a buffer zone around the bubble. The gradient serves to push objects back into the bubble.

To capture this behavior, Locus defines a utility function for each object based on the distance δ from a specific location to the object d's home location $d.\chi$. Given a bubble size of γ, the utility function maps γ and δ into a value in $[0.0, 1.0]$. One function that does so is a parameterized sigmoid function:

$$u(\delta) = 1 - \frac{1}{\left(1 - e^{\alpha \cdot (\beta - \delta)}\right)} \tag{4.1}$$

In Equation 4.1, β is the point where utility is 0.5 which determines how far away from the bubble center utility will start to drop off. α determines the rate at which the utility changes. The shape of Equation 4.1 is shown in Figure 4.2 for a given α and β. As can be seen, the choices of α and β determine the bubble size γ and width of the buffer zone ω. γ is the point where utility starts to drop, roughly, $u(\gamma) = .99$. The buffer zone extends from γ to the point where utility nears 0, so $u(\gamma + \omega) = .01$. α determines the size of ω by controlling how fast the function moves from .99 to .01. By solving Equation 4.1 for a fixed α and β for $u(\delta') = 0.99$ and $u(\delta'') = 0.01$, then $\omega = \delta'' - \delta'$. Given ω, the value of γ is $\gamma = \beta - \omega/2$. By precomputing the different ω values from specific settings of α, an appropriate α and β can be chosen for each data object to maintain the desired bubble shape.

The value of bubble size γ plays an important part on the behavior of the system. If the bubble size is too small, it is possible that no nodes will be in the bubble and those objects will be quickly dropped.

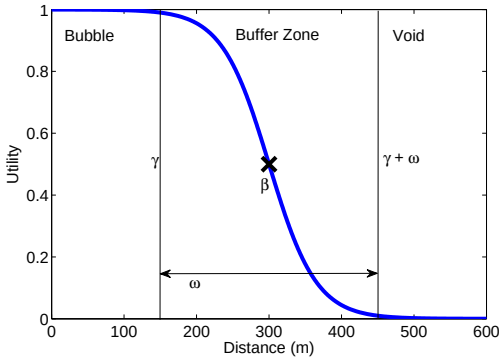


Figure 4.2: Equation 4.1 with $\gamma = 150$ and $\omega = 300$. This is generated from parameters $\beta = 300$ and $\alpha = 0.307$.

At the same time, if the bubble size is too big then objects will start to drift too far away from their home location, making look-up difficult. Therefore, the bubble size should be large enough so that it should always contain at least one node. Ultimately, the correct bubble size depends on the density of the network at the specific location where d is created and the accuracy of location measurement available at nodes. Based on these parameters, γ should be set dynamically for each object. Currently, γ is set statically. Dynamically adjusting γ based on network conditions is an important next step for future work.

4.3.2 Object Prioritization

When two nodes meet, each must decide using Equation 4.1 which objects to replicate to the other. Each node will attempt to replicate as many objects as possible to the other node during the limited contact opportunity. Each node attempts to select objects to replicate that will bring the system the most utility. There are three scenarios in which it is especially beneficial to copy an object.

First, if the node is currently inside the bubble with a high utility at its current location, copying the object to surrounding nodes increases the probability of a replica remaining in the bubble. Second, if a node is outside the bubble or the buffer zone, the object should be replicated to a peer currently in the bubble. Any node that has higher utility at its current position than the current node should receive an object replica. Finally, even if two nodes are outside the object bubble, if the peer node is moving into the bubble in the near future, the local node should copy the object to the peer so that the object is carried back into its bubble. If the peer node will have a high utility for the object at a future location, then the object should be sent to the peer.

Any of these three scenarios is beneficial to the goal of maintaining objects within their bubbles. Therefore, Locus computes the utility of each object at each location in the above three scenarios. The object with the highest utility in any scenario is sent first to the peer where that utility occurs. Formally, for

each object d at node n with connected peer p, the priority is the maximum value from applying Equation 4.1 in each of the above three scenarios. Formally, each scenario is defined in terms of $u(\delta)$ where δ is the distance between a node's position and the object's home location $d.\chi$ computed by the distance function $dist()$.

$$prio_1(d, n, p) = u(dist(n.pos_{current}, d.\chi)) \tag{4.2}$$

$$prio_2(d, n, p) = u(dist(p.pos_{current}, d.\chi)) \tag{4.3}$$

$$prio_3(d, n, p) = \epsilon \cdot u(dist(p.pos_{future}, d.\chi)) \tag{4.4}$$

The final priority is the maximum of the three functions:

$$prio(d, n, p) = max(prio_1(d, n, p), prio_2(d, n, p), prio_3(d, n, p)) \tag{4.5}$$

Estimating the future position of a node can be done simply by storing a small history of previous locations and computing the current device trajectory. While simple, this method is prone to prediction errors. If available, more accurate information can be gathered from GPS units which might indicate the planned route of a device or even know the device's target destination. To account for the varying degrees of error in prediction, Equation 4.4 is scaled by a factor $\epsilon \in [0.0, 1.0]$ that can be adjusted depending on the prediction method used. In the case that users are unwilling to share their future location for privacy reasons, $\epsilon = 0.0$ to suppress using future knowledge.

4.3.3 Dropping Objects

It is never bad to storage objects where they have low utility if storage has not reached its capacity. However, when storage space is completely consumed, some objects must be removed. To select objects to be removed, Locus finds the object in the local buffer with lowest priority. In this case, a node can only consider its local information. If a node is far away from an object's home location and will not move into the object's bubble, that object should be deleted to make room for local objects. Therefore, an object d is prioritized by the node n using Equation 4.2 where $p = null$ and also considering the node's future position:

$$prio_4(d, n, p) = \epsilon \cdot u(dist(n.pos_{future}, d.\chi)) \tag{4.6}$$

Objects with the lowest priority are dropped first. If the utility of any objects is the same, i.e. if both objects were generated at the same location or both objects are out of the object bubble buffer zone (ie. $u(\delta) = 0.0$), then the creation time $d.\tau$ is used as a tie-breaker. Older objects are dropped first and correspondingly newer objects are forwarded first.

4.4 Data Access

To access data in Locus, nodes forward query messages to a specific target location looking for matching data objects along the way. The location for which a query is interested is the *target location*. Nodes can match objects to queries based on two parameters: location and creation time. Each query message has in addition to the target location, a target range which specifies the area around the target location in which the query is interested. Any objects whose home location is within the target range of the target location match the query. The query also can specify a time range to match objects. Only objects which have a creation time within the specified time range can match the query. A query message q is defined to be the tuple $(\chi, \rho, \tau_{start}, \tau_{end})$ where χ is the target location of q, ρ is the range, τ_{start} is the oldest acceptable creation time of data and τ_{end} is the newest acceptable creation time for data. Either or both of τ_{start}, τ_{end} can be set to infinite to indicate an unbounded time. If both are infinite, then there is no time constraint for matching objects.

As queries are forwarded toward the target area, any intermediary node can respond to the query if it contains an object whose home location matches the query. As the query nears the location, it will encounter bubbles for data based around the target location. Within the bubbles the probability to find matching data is higher. Once a matching object is found, a response is forwarded back through the network to the original node's current location. Response messages contain a target location χ and range ρ to find the original node and the value ζ of the matched data, $r = (\chi, \rho, \zeta)$

4.4.1 Forwarding Messages

The challenge in forwarding messages comes from the fact that messages share the same limited contact opportunity as data objects. If message forwarding is too aggressive, the transfer of messages will consume all of the bandwidth and prevent objects from being replicated to their bubble. On the other hand, if message forwarding is too passive, messages will be starved and look-up success rate will be low.

Query and response messages can be thought of as data objects where the home location is not where the message was created, but instead where the message is destined. Because of the similarity between

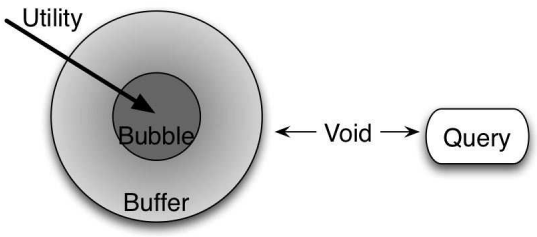


Figure 4.3: Query and response messages are generated far from the target location bubble meaning they have to cross a void of 0 utility to reach the destination.

objects and messages, it seems natural to use the same utility functions for sorting messages as are used to replicate data objects. However, in most cases queries will be generated far from the target location. According to Equation 4.1 the utility at distances outside of the bubble is 0 or close to it. If Equation 4.5 is used to prioritize query messages, those messages will have lowest priority and never be forwarded.

To move messages through the void where location utility is 0 (see Figure 4.3), a new priority function is needed. The aim of Locus's focusing is to move messages as quickly as possible toward their target location. If the message is very far away, it is unlikely that a connected path to the target location exists, therefore, at each encounter, the message should be given to the node that can carry it closest to the target location. Existing approaches use complicated models that require deep knowledge of the map on which devices are moving, the traffic levels on different streets and the destinations nodes are heading towards [46, 130, 136]. Locus uses a more probabilistic approach that is better suited for ICNs where some nodes do not move on predictable maps and which integrates better with the replicating of data objects.

To decide if a message should be passed to a different node and if so, to which node to forward it, nodes compare the future location of connected peers to their own. The message is forwarded to whatever node has a future location closest to the target location. To ensure that these messages have a high enough priority to compete for bandwidth, if Equation 4.5 is 0 for a given message, the message is given a large priority value, $\Upsilon < 1.0$. In this way, messages that are far from the target location will quickly move towards the location. Once a message reaches the target location, the same priorities used for object replication in Equation 4.5 can be used when $prio(q, n, p) > 0$.

To achieve a balance between forwarding messages and replicating objects, Υ is large enough that messages have high priority, but not too high to starve out data objects. In this work, $\Upsilon = 0.95$. Because multiple peers might be moving closer to the target location than the current node, the final priority for a message is weighted according to each peer's distance with the value $1/dist(n.pos_{future}, q.\chi)$. The message is forwarded to the peer for which the priority is highest, or kept at the local node if it has the highest pri-

ority.

$$prio_{forward}(q,n,p) = \begin{cases} \Upsilon + \frac{1}{dist(p.pos_{future}, q.\chi} & dist(p.pos_{future}, q.\chi < dist(n.pos_{future}, q.\chi) \\ -1 & dist >= peer.dist \end{cases} \tag{4.7}$$

4.4.2 Containing Response Generation

When a node matches one of its locally stored objects to a query, it generates a response message. Because the response messages contain the matching data object, they are much larger than query messages. If too many response messages are created for a specific query, then the channel will be congested. On the other hand, a single response message might not have high enough probability to reach the destination. To manage the load of sending responses and queries while keeping a high query satisfaction rate, Locus seeds queries messages into the network using an approach similar to quota-based SCF protocols [105, 121, 122]. Each query message is given a seeding quota $q.quota$. Once a query message is matched, that copy is deleted from the network. At most, $quota$ responses will be generated for each query.

Initially, as query messages are copied between nodes, the quota is divided between the two nodes following a binary distribution that achieves a faster spread [121]. Messages with more quota left are given a higher priority so that so that multiple paths are explored [105] by each query message. By moving multiple copies toward the target area, the chance of finding matching data objects increases significantly. The priority for a message based on the remaining quota $q.quota$ and initial starting quota for each message $initialQuota$ is:

$$priority_{quota}(q,n,p) = q.quota/initialQuota \tag{4.8}$$

The final priority for a query message then is the max of Equation 4.7 and Equation 4.8. If the local quota for a query message is 1 and the value of Equation 4.7 is less than 0, the message is not forwarded.

4.4.3 Sending Responses

By setting the target location of a response as the location of the querying node, response messages can be returned to the querying node using the same priority functions as for queries. The challenge is that nodes are constantly moving and a node's current location is difficult to know. To overcome this challenge, Locus does several things. First, the querying node includes its current location in the query message when the message is first generated. Second, the node also includes its current trajectory in the query

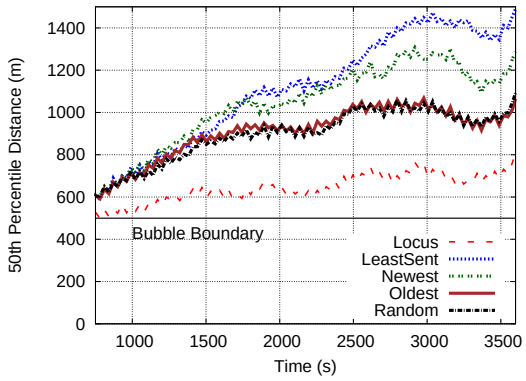


Figure 4.4: The change over time in median distance of all objects from their home location.

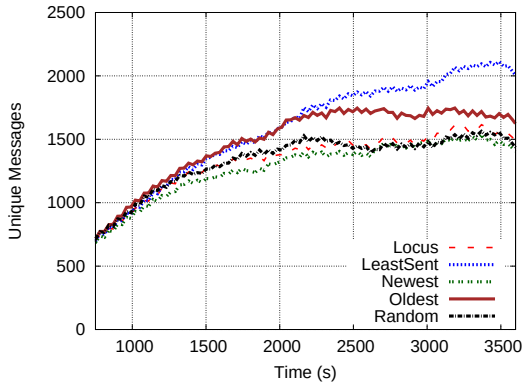


Figure 4.5: The change over time in number of unique messages in the network.

message. Third, when a node generates a response message for a particular query, it uses the enquirer's original location, trajectory and the age of the query to estimate the querying node's current location. Based on the age of the query, the target range is adjusted to account for increasing error in estimating the target location. Finally, when a response message reaches the target area, it is spread using Epidemic style flooding [128].

4.5 Simulation Evaluation

The goal of the evaluation is to show that through its location-based policies Locus can i) keep data objects within their bubble and ii) that having data close to a known location improves the probability of queries finding the data. Locus is evaluated in a simulated environment of instrumented vehicles. This environment presents physical distances and device speeds similar to what is expected in real deployments. However, due to the limitation of simulation, it is difficult to create scenarios with the node density similar to what is expected in real life. In the simulation scenarios, 150 vehicles move in a 5km x 5km area fol-

lowing movement over a map of Chicago generated using the VanetMobiSim [73] vehicle simulator. While this density of devices is far below that expected in real deployments, the results obtained in this environment will actually be far worse than in a real deployment because of the sparseness of the network. To ensure that object bubbles are covered by at least one node, the bubble size is set larger than normal with $\alpha = 0.307$ and $\beta = 650$ resulting in a buffer zone of 300m and a bubble size of 500m. The drawback is that the number of individual objects each node has to store is higher than normal. The traces were used to control mobility in the ONE simulator [82] to simulate data traffic.

The second challenge for evaluating Locus is to find a suitable comparison protocol. Existing SCF forwarding protocols cannot be directly applied for storing and replicating data because they are destination-focused for end-to-end forwarding of messages rather then persistent storage. To compare Locus against other possible overlay implementations, different message-based replication policies from ICN environments are used to store data and combined with basic Epidemic forwarding [128]. The different replication policies affect the way data is stored in the network and queries and responses are flooded to maximize the search. Several replication strategies are taken from literature, including LeastSent, which always replicates the objects that have been copied the least, Oldest, which always replicates the oldest objects, Newest, which replicates the newest objects first and Random, which randomly replicates objects.

In each scenario, every node generates new data 1KB objects every 10 seconds, mimicking a sensor application that profiles traffic or environmental conditions. Each node has a buffer big enough to hold the objects generated from three nodes over the length of the simulation. Transmission rate is 2Mbps and nodes move along streets at speeds between 10mph and 55mph.

4.5.1 Data Storage

The effectiveness of Locus's location-based object replication policy is evaluated by comparing it against the other replication policies. The goal of Locus's replication policy is to keep data as close to the home location as possible to support location-targeted queries. Because Locus prioritizes data objects for replication based on their distance from the home location, the Locus policy should be significantly better at keeping data within the bubble. Other replication policies do not take distance into account and therefore allow data to drift throughout the network. This fact is illustrated in Figure 4.4, which shows the change over time in the median distance of each object to its home location. Locus's prioritization functions successfully keep objects near or within their bubbles.

The trade-off in Locus's replication policy is that it does not focus on preserving object lifetime. The result is that some objects are dropped from the system earlier than with other policies. Figure 4.5 shows

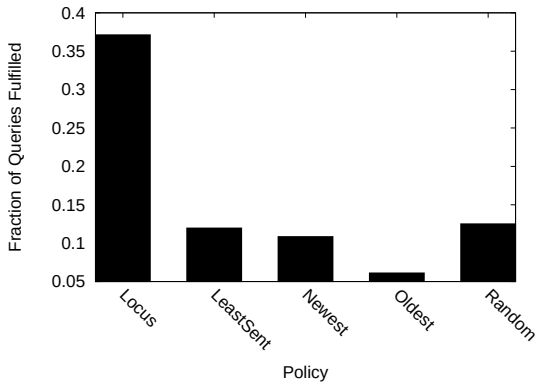


Figure 4.6: Overall success rate of different policies to satisfy queries.

that Locus tends to make more copies of fewer objects in order to keep the objects close to the home location. On the other hand, LeastSent attempts to ensure that each object has an equal number of copies in the network. The sacrifice is that objects are spread throughout the network and are not necessarily close to the home location.

4.5.2 Data Access

The replication policy is useful only in how well it can preserve objects to be found by application queries. To evaluate the query satisfaction rate using different policies, the query satisfaction rate using different replication policies was measured. Queries were generated by randomly selected nodes at a fixed rate network-wide. For each new query the target location was selected randomly from a location on a street in the space. This ensures that it is possible for a data object to have been generated in range of the target location. The query range was set to 200m.

In the first experiment, the overall query success rate was measured. Each query was given an infinite time range to focus on the interaction between message forwarding and data copying policies. As can be seen in Figure 4.6, Locus's location-based policies were far more effective at connecting users to data than other policies. This is a combination of Locus's policies of keeping objects in bubbles and the location-augmented forwarding techniques. Despite the high number of unique messages preserved in the system by LeastSent and Oldest policies, the messages are scattered around the network and difficult to find, even using Epidemic-style flooding, resulting in lower query success rate.

However, as illustrated in Figure 4.5 Locus sacrifices object lifetime in favor of proximity. The trade-off means that it is harder to find historical data. To quantify the effect, a simulation was run in which the simulation is divided into 5 epochs of 12 minutes each. Each query is generated with a time range lim-

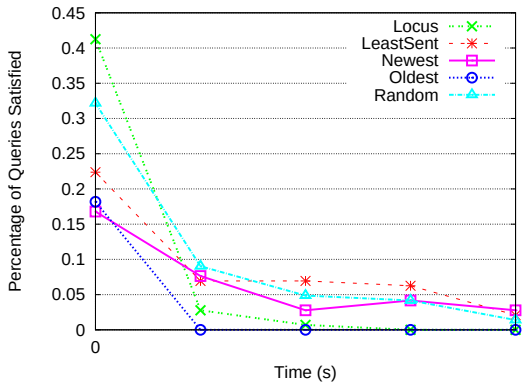


Figure 4.7: Success rates in satisfying historical queries.

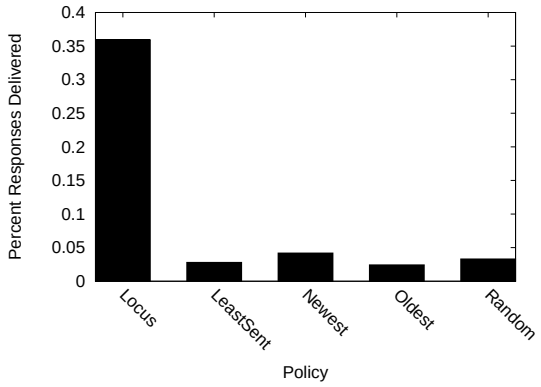


Figure 4.8: Locus achieves higher query satisfaction rate through improved message forwarding, seen here in percentage of response messages actually delivered.

ited to the first epoch. As the simulation progresses it becomes more and more difficult to find the original data. As shown in Figure 4.7 Locus's policies favor recent data over historical data, leading to lower query success rate for historic data.

The benefits of Locus come from a synergy between the bubble oriented object replication and the location-augment message forwarding. To demonstrate the importance of the message forwarding component, the effectiveness of delivering responses after a query was matched was measured. Because of the node mobility, this is a difficult but important phase of the entire process. As can be seen from Figure 4.8, a big part of Locus's improved query response rate comes from its unique response forwarding policies. By focusing a message to a specific area before spreading it, Locus can greatly increase the probability of finding the original node.

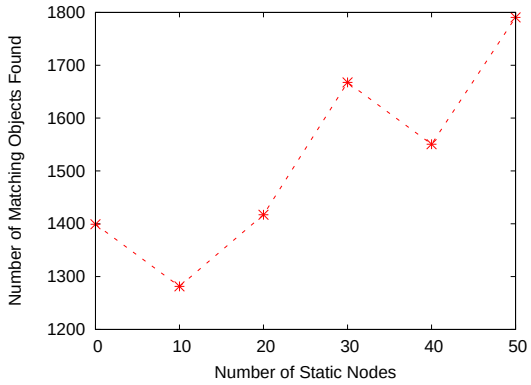


Figure 4.9: Total number of matching objects found in response to initiated queries with different numbers of static nodes.

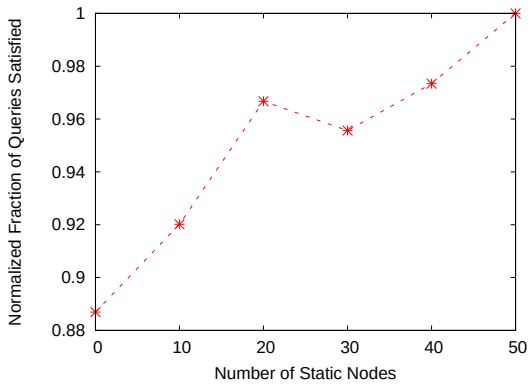


Figure 4.10: Normalized network-wide query satisfaction rate with different number of static nodes.

4.5.3 Utilizing Static Nodes

So far, the simulations have assumed that every node is constantly moving. However, in reality a city environment will be filled with parked cars and other stationary nodes that can assist in Locus behavior. Static nodes help maintain an object's bubble, acting as a semi-permanent storage for a given area and also improve message forwarding, acting as throw boxes [31] to increase connectivity. These two benefits are seen in the number of queries that successfully found object matches (shown in Figure 4.9) as well as the total query satisfaction rate (shown in Figure 4.10). As the number of static nodes increases, the bubble fidelity improves, increasing the number of matched objects for each query (Figure 4.9). This increased match rate combined with improved forwarding connectivity for queries and responses leads to increasing query satisfaction rates (Figure 4.10).

4.6 Future Work

One important factor in the performance of Locus is the size of bubble used to store data. If the bubble is too big, excessive network resources will be consumed. On the other hand, if the bubble is too small objects will quickly die. The ideal bubble size depends on the density of devices in the given area. The bubble should cover at least a few devices to ensure high probability for objects to survive. As the density of the network changes, the bubble should change as well. An interesting avenue for future research is to explore how to detect surrounding network conditions in order to dynamically adjust the bubble size.

In addition, the utility function used in the current implementation has a uniform shape for all data objects. However, the utility function could have very different shapes, depending on which kind of data they refer to and how far away from the home location the data is relevant. Since different events and environments will result in different levels of interest, the utility functions can change dynamically to capture different application specifications and current user interests. One application of allowing dynamic utility functions is to enable caching of objects at multiple locations in the network in addition to the home location. Utility functions can be defined that push objects to different locations in the network, allowing the objects to live nearer where queries are originating, reducing the forwarding needed to satisfy queries.

4.7 Conclusion

The sensor and user-created data generated on end-user devices has the potential to enable a new class of live, location-based services for commuters, pedestrians and other mobile users. However, the large volumes of data these devices will generate cannot be supported by existing centralized approaches. Instead, a decentralized data overlay was designed that runs on top of the mobile devices themselves. Because data producers and consumers are the same in this network, keeping the data near those devices brings greater utility to the network.

Locus introduces the novel concept of "bubbles of knowledge" to keep data about a specific location near that location. By using location information, Locus can achieve more efficient data storage approaches and improve the reliability of storage and the success rate of queries. As more users join the network, the benefits of Locus will be amplified due to the higher density of the network, increasing both the performance and value of the overlay as the system grows.

Chapter 5

Inter-network Communication

To completely satisfy end-user demands for mobile data access, the UCN must connect to the Internet. Such inter-network communication is not new; it forms the basis of the current Internet. In the current Internet environment, operators of different networks form agreements with each other either to purchase or to peer traffic between their two networks. The operators then install gateways within their network to connect to the external network. Internally, traffic destined for external networks is forwarded to the gateways and then into other networks. Externally, the Border Gateway Protocol (BGP) [94] is used to enforce the inter-network policies. Access control is easy to manage since entire networks are represented by single operators. This combination of internal gateways and BGP configuration manages the inter-network connectivity for existing networks.

Although this management approach has been successful for existing networks, the nature of the UCN prohibits using the same management technique. Unlike existing networks which are managed and controlled, built at a cost to the network operator for the purpose of making a profit, the UCN is not managed. No single operator is responsible for its deployment or maintenance. Without a clear owner in the UCN, arranging an agreement with other networks for Internet access is impossible. Similarly, there is no support for deploying and maintaining dedicated gateways in the UCN to carry all inter-network traffic. Even if such obstacles could be overcome, the existing network operators will not tolerate the use of their network by UCN clients without some access control and traffic management, which is typically enforced by BGP. To ensure the participation of external network operators and provide Internet access to UCN clients, new approaches are needed to manage the interactions between the UCN and external networks. There are two network management tasks that are essential in this case: connectivity management and authentication.

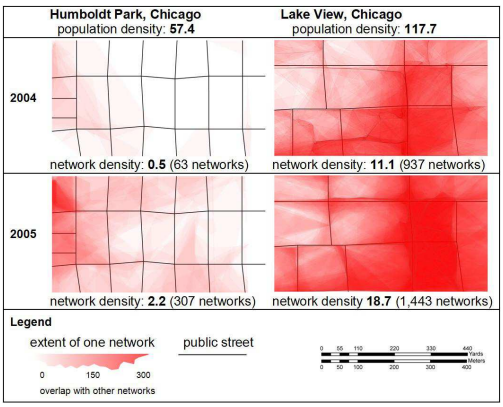


Figure 5.1: IEEE 802.11b/g diffusion in two typical Chicago neighborhoods: a community park and residential area. Densities in millions per decimal degree of latitude and longitude.

5.1 Connectivity Management

Connecting the UCN to other external networks is hindered by the lack of applicable network management services. Connectivity to external networks is usually handled by making some form of agreement, using BGP [94] to enforce the agreement, and deploying gateways in the internal network as bridges into the peer networks. This approach will not work in the UCN where there is no network operator to negotiate contracts and no affordable way to deploy or maintain gateways and BGP servers to cover all areas where the UCN might exist. Instead, user-created networks require a fundamentally new approach to bridge external networks. An emerging approach for this scenario is to create networks of shared Wi-Fi access points (APs). Following the peer-to-peer spirit, individual users share their subscribed broadband access over their private wireless routers. This approach has not only been advocated in this thesis, but also in both academia (Extended HotSpots [5], OBAN [50], P2PWNC [51]) and industry (ABitCool [1], Fon [7]).

Existing methods of managing connectivity by deploying internal gateways and BGP do not work because of the already mentioned cost and maintenance requirements. Because there is no central operator for the UCN, the connectivity between UCN and external networks must be managed by the individual end-users themselves. To provide access to external networks, end-users share their idle broadband or other wide-area connection with UCN clients through a wireless gateway. Most often, the wireless gateways are Wi-Fi APs deployed in private homes and made publicly accessible. However, the gateway could also be a laptop computer, mobile phone or any device that has wide-area backhaul and can connect to the UCN using unlicensed wireless technologies. Although unplanned and spontaneous, opportunistic connection through these APs can provide good coverage in many urban/suburban areas. As shown in Figure 5.1, the coverage between 2004 and 2005 for two regions in Chicago shows that even five years ago

coverage was extensive. In most cases, the networks to which the gateways are attached are subscribed broadband such as Cable or DSL. However, backhaul could also come from different campus networks such as universities, apartments or corporations, for example. By leaving the management of the access points under the control of the owner, the gateway can ensure that the UCN complies with the owner's individual policy agreement or subscription for the backhaul. Each gateway is managed and operated independently from the others.

Relying on shared end-user APs ensures ISP policies are enforced, but it also results in unreliable, low throughput and high delay communication for the UCN clients. Because AP owners usually pay for broadband access, they desire that their own traffic is protected from competing traffic from the UCN and that the UCN traffic does not cost them extra money. To do so, traffic shaping is employed on the shared gateways. With a potentially high number of clients at each gateway, this means that there is limited bandwidth to be split among many users, leading to poor performance in terms of throughput and latency for the UCN clients. The clients themselves can lessen the problem by attempting to associate to the gateway with the best performance, but the conditions on each gateway can be variable and unpredictable. Instead, the abundance and accessibility of 802.11 wireless networks suggest that a promising approach to improve Internet access is for clients to aggregate Internet access from multiple gateways simultaneously.

In order to manage the limited bandwidth resources available to UCN clients, a new framework for end-host multihoming is developed in Chapter 6. The framework uses locally-learned end-user traffic patterns to predictively measure and schedule new TCP flows on the available access points. Without relying on any end-to-end solutions or input from other nodes, high-reliability, high-throughput and low-delay Internet access can be achieved by each UCN client.

5.2 Authentication

The other side of network management comes from the external network operators' and gateway operators' viewpoints. Operators and owners are concerned with managing who can connect to their network and what resources those users can consume. By sharing their gateway, owners open their network to potentially anybody who is nearby. As a foundation for security and incentive enforcement as well as other management tasks, owners and ISPs require authentication of UCN clients to reveal both the client access privileges and identities.

Because of the vast scale in size and number of clients in the UCN, providing authentication cannot be left to the individual owners of gateways. For the majority of end users, the task is technically too chal-

lenging and time consuming. In addition, the use of ISP resources demands the cooperation of the ISP operators. To be deployable in the UCN, authentication credentials must be controlled by a third party trusted by UCN clients, gateway owners and network operators. In addition, as Wi-Fi users are becoming more security aware, the demand for encrypted wireless communication is high. Therefore, the deployed authentication framework should enable both access to the backhaul network as well as encrypted wireless communication.

Most existing authentication solutions rely on central servers to handle authentication. Captive portals and the 802.11i suite of protocols tunnel authentication requests to a central server that stores all relevant credentials and verifies the clients. Other protocols exist to reduce the load on the central server by only checking the revocation status of a certificate [104]. Due to the unreliable connections in the UCN, communication with central servers faces high failure rate, especially since most protocols are not reliable. In addition, due to the large size of the UCN a central server might be located far from where authentication is happening or overloaded with potentially millions of requests, which leads to high communication and processing delays. For mobile users whose connectivity periods might only last a few seconds, the extra delay for authenticating is intolerable. One approach to reduce delay is to deploy servers closer to the edge of the network to act as authentication proxies [50]. Although such an approach can reduce communication delay, in order to service the entire UCN deployment, the installation and maintenance costs are large. Especially considering the lack of strong business model, it would seem that there would be no incentive for a third party to invest large sums of money up-front before the UCN has been proven successful. What is needed is a new deployment approach for authentication between the UCN and backhaul networks that runs on end-user devices themselves, has low delay and overhead and yet is still centrally managed.

In response to these challenges, a new authentication framework is presented in Chapter 7 that enables low-latency and reliable authentication by pushing the authentication servers to the edge of the network at end-user gateways. Certificate-based authentication allows for offline authentication between devices. Location-based mechanisms are used to distribute certificate revocation lists, reducing the storage and computation overhead and reduce latency of the authentication procedure. Devices opportunistically verify the freshness of the presented credentials using as much information as is available to them.

Chapter 6

Flow Scheduling for Inter-network Connection Management

Providing Internet access from the UCN is difficult because there is no central operator to negotiate and maintain access for the global network. Instead, the UCN pools shared broadband access from many individual end-users. The end-users negotiate their own access from different ISPs and individually manage their access points to ensure that ISP policies are enforced. This collaborative Internet access infrastructure avoids the need for a central operator, but it also means that the connectivity available to each UCN client is limited. Because of contention, traffic shaping and poor wireless connectivity, the throughput, delay and reliability of Internet connectivity all suffer. End-users can overcome these limitations if they can pool all available bandwidth from different APs. In this chapter, a bandwidth aggregation framework is developed that delivers high-throughput, low-delay and high-reliability Internet access to UCN clients through predictive flow scheduling based on local information only. Deployment in a residential testbed confirms the benefits of the framework.

6.1 The Bandwidth Constraint in Shared Open Wi-Fi Networks

Managed Internet access can be made available to user-created networks through private shared 802.11 access points and Internet-connected mobile devices. However, the number of clients attempting to access the Internet will likely always outnumber the gateways providing access, creating a bandwidth bottleneck. The bottleneck is made worse by the fact that any traffic flowing over a gateway will have lower precedence and service than traffic from the gateway's owner. Additionally, outages in the ISPs can leave users or communities disconnected for hours or even days [1]. Given that the internal bandwidth of the user-created network is likely higher than the bandwidth at a single gateway, the outgoing bandwidth can become a severe bottleneck for the UCN.

At the same time, the density of 802.11 networks connected to broadband connections is increasing.

[1] Typical broadband ISPs do not guarantee the availability of Internet service instead offering a refund typically only after 24 hours outage. See http://www.insightbb.com/terms_conditions/ and http://worldnet.att.net/general-info/terms-dsl-data.html

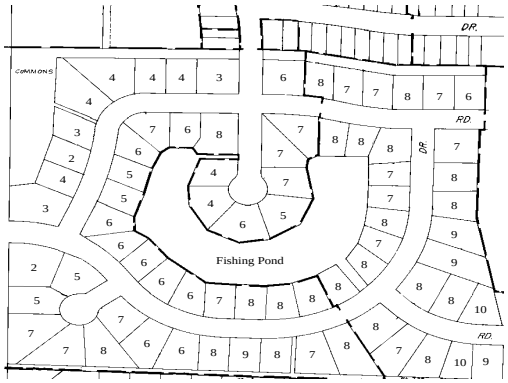


Figure 6.1: Number of per-household 802.11 networks detected in a residential area.

By June 2007, over 82.2% of American Internet users and 53% of all US households subscribed to some form of broadband service [8]. The density of broadband users is already high. Second, broadband access in residential areas has diversified to the point that many residents have a choice from several broadband ISPs serving the same area with DSL, Cable, Satellite, or even WiMAX. Third, there has been wide-scale deployment of IEEE 802.11 (802.11 for short) home wireless networks. Given the communication range of 802.11 devices, it is common for residents to receive signals from their neighbors' wireless routers. Figure 6.1 shows the map of the subdivision where one of the authors lives. The area is served by three ISPs: two DSL providers and one Cable provider. As can be seen from the map, while using a laptop with integrated 802.11 interface and internal antenna from the driveway of each house, an average number of 6.4 802.11 networks were detected at each residence. Finally, the broadband Internet access is always on but seriously under-utilized. Recent surveys show that world-wide the busiest residential users are online for 27-39 hours a month on average, or only about 6-10 hours a week [3].

Projects such as FON [7] and Seattle Wireless [13], for example, demonstrate a willingness amongst consumers to participate in collaborative networks. By connecting *simultaneously* to his own broadband and one or more of his neighbor's wireless networks, a residential user can achieve higher bandwidth and higher connection resiliency. More importantly, no additional monetary cost beyond the existing broadband subscription and perhaps a second 802.11 wireless card is required.

What is lacking is a comprehensive framework to enable and guide the sharing. To reduce cost and facilitate ease of deployment, the solution must be deployable within a single home without relying on any external mechanism. This chapter presents the design, implementation, and evaluation of a framework for Predictive End-host Reliable Multihoming, or PERM. PERM is a system that enables managed collaborative Internet access in UCN areas.

Because PERM is deployed at the edge of the network by end-users, it must achieve its scheduling

without requiring changes in the Internet infrastructure. Previous proposed advanced routing schemes, transport protocols or bandwidth aggregation techniques cannot be deployed in the PERM scenario because of this limitation. Therefore, PERM performs *flow* scheduling across the available Internet connections [2], similar to enterprise multihoming. Current enterprise multihoming flow schedulers operate either by load balancing [63] or latency scheduling [24]. The existing algorithms work well in the enterprise where there is high volume of traffic; however, end-host multihoming schedules only an individual user's flows rather than the aggregated traffic from an entire enterprise. The existing scheduling approaches are either dependent on high volumes of traffic for building accurate link profiles or schedule too coarse-grained to be efficient in the sparse residential setting, as later experiments reveal.

PERM exploits its proximity to the end-user to improve scheduling effectiveness by adopting automated on-line learning of the end-user's traffic patterns to make the best match between flows and available Internet connections. From the Nov. 2003 - Feb. 2004 network traffic traces collected in *residential* buildings at Dartmouth College [87] The networking traffic of the top 100 busiest users is analyzed. Discoveries for individual users in the residential wireless traces include a strong temporal correlation among the sets of destination addresses a user visits at adjacent time intervals and the long-range dependency of the traffic volume when a user communicates with a specific remote host. Based on these findings, on-line prediction algorithms are designed to infer the set of remote IP addresses with which a user will most likely communicate in the next time interval, as well as the traffic volume associated with each connection. With those models and prediction algorithms, *scalable pre-probing* and *hybrid flow scheduling* are designed to achieve zero-latency, traffic-aware, non-preemptive scheduling for end-host multihoming. The effectiveness of the algorithms are evaluated in an implementation for Linux, deployed in a live testbed. Evaluation results show the performance improvement of PERM's hybrid scheduler over other schedulers and the significant benefit of collaborative Internet access in general.

6.2 Multihoming and End-user Networks

Much prior research has examined effective approaches for utilizing multiple Internet connections. Route selection is probably the closest to PERM in that the main function of a PERM scheduler at the client side is to route flows across different neighbors' ISPs. Existing route selection is usually achieved by either source routing [38, 62], proxy routing [28], or overlay routing [27, 114]. However, the application of those mechanisms in a PERM-like setting will require either advanced support from Internet routers (IP source routing), the deployment of routing proxies beyond the bottleneck Internet access links (proxy routing),

[2] In the implementation a flow is identified by a [SrcIP, SrcPort, DstIP, DstPort] tuple.

or intermediary Internet infrastructure (overlay routing). It is unlikely that any of those resources will be freely available for each residential user in the near future. PERM on the other hand relies only on local resources already available to the end-user.

Similar drawbacks stand in the way of deploying previous bandwidth aggregation techniques. Link layer bandwidth aggregation coordinates multiple underlying radios to present a virtual interface to higher levels [19, 20, 120]. These techniques do not apply in PERM where the bottleneck Internet access links potentially belong to different ISPs which will drop packets with foreign IP source addresses. Transport layer aggregation has also been proposed [70, 95, 54]. However, it requires modification at both ends of the connection. Other approaches rely on the deployment of intermediate proxies (e.g. aggregation proxy [115], and Mobile-IP agent [42]). The PERM user is generally unable to deploy such proxies. By scheduling at the flow level PERM avoids these constraints, but loses some efficiency.

In contrast to the above approaches, multihoming operates completely locally and is the approach adopted in PERM. Multihoming was first shown to achieve more than 40% improvement using multiple ISPs, assuming perfect knowledge about the providers and ability to change routes arbitrarily [22]. Following work [23, 127] revealed that multihoming could achieve the same reliability and end-to-end performance as overlay routing. Several different multihoming schedulers have been presented in the literature based on simple load balancing and hashing techniques [63] as well as active and passive probing of the available links [24]. One drawback of these approaches is that they are designed to optimize the aggregate traffic performance from the entire enterprise. The sparse level of traffic in the end-host prevents effective passive probing, as there is not enough traffic to the target destination, and also eliminates post-facto active probing, which relies on the assumption that different users will soon visit the same site. Although probing-based schedulers are able to learn finer-grained knowledge of each connection, latency-based scheduling is less effective when throughput becomes the dominating factor [63]. PERM adopts a hybrid latency/load-balancing scheduling scheme that adapts based on the expected volumes of individual flows. It performs uniformly well for both latency-sensitive light-volume flows and throughput-sensitive heavy-volume flows.

PERM's predictive scheduling mechanisms are built on individual user Internet usage models. Early studies of the Dartmouth College 802.11 WLAN wireless traces [87] analyzed the temporal and spatial distributions of all users, their networked applications, and their traffic aggregate [69, 86]. In this study, the individual user behavior of the busiest users is analyzed to find models for *individual* behavior. In other analysis, it has been shown that, for HTTP traffic specifically, the information accessed by HTTP requests holds the property of spatial locality [43].

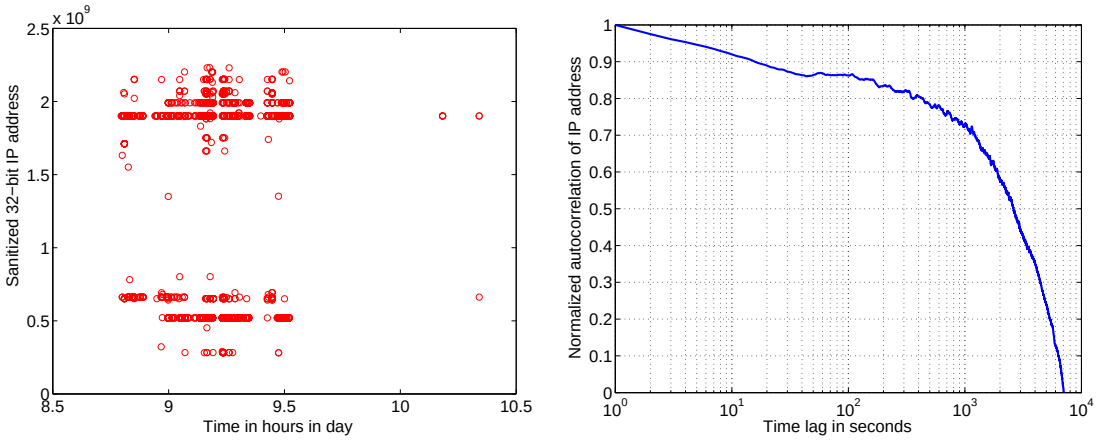
6.3 Predictive End-host Reliable Multihoming

PERM enables collaborative Internet access through *high-performance flow scheduling*, *incentive and flow management* and *security and privacy enforcement*. A compliant user subscribes to a certain form of broadband Internet access, e.g., DSL or Cable. The user shares his Internet access among his own client devices (home clients) and devices in the UCN (foreign clients) through a high-speed wireless router as the gateway to his ISP network. The wireless gateway does traffic shaping to control the flows initiated by the owner and neighbors.

A scheduler runs locally on each client to appropriately direct a user's flows to the best gateway for that flow. It is assumed that clients can simultaneously connect to multiple gateways. Multinet [41] allows a single radio to connect to two SSIDs at the same time. Additionally, most laptops come with two Ethernet NICs: 802.11 and Ethernet. If an Ethernet connection is not possible, an additional wireless USB or PCMCA card can be purchased for under $100 and even shared between different computers in the home if necessary. The issue of selecting which neighboring gateway to connect to is not dealt with. It is up to the user to connect his wireless interfaces independently. A system like Divert [102] could be used to facilitate the automatic selection of and association to access points. Any such schemes could easily be added into PERM.

6.3.1 High-performance flow scheduling

Because PERM is constrained to the edge of the network the scheduler must schedule flows instead of packets. There are two different constraints on the performance of Internet flows depending on the volume of the flow. A flow is defined as either a light-volume flow or heavy-volume flow according to the flow's limiting factor. The performance of light-volume flows is dominated by the round-trip time (RTT) to the destination because the flows will never saturate the link during TCP slow start. On the other hand, heavy-volume flows that generate more packets will be constrained by the capacity of the connecting link. PERM employs a hybrid scheduler to perform high-performance non-preemptive flow scheduling based on these two different metrics, RTT and link capacity. PERM exploits its proximity to the end user to analyze end-hosts' traffic patterns. The current RTT on each link is maintained by pre-probing the destination, eliminating the latency of post-facto probing after the connection request is made. Heavy-volume flows are scheduled based on their predicted volume and the current load on each access link. Since PERM cannot probe all remote Internet hosts, the following two questions have to be answered for scalable pre-probing and hybrid flow scheduling: (1) What is the set of remote IP addresses that a user is most likely to communicate to within the next time interval and is it predictable? (2) What is the distribution of the



(a) IP addresses visited over time

(b) Autocorrelation of the IP addresses

Figure 6.2: Remote IP connection behavior in Dartmouth traces.

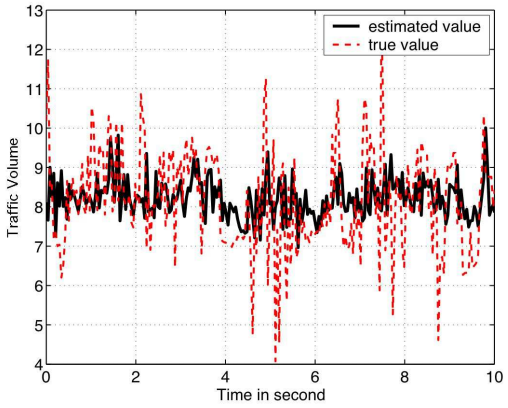


Figure 6.3: Predicted versus actual volume.

traffic volume associated with a remote host and is it predictable?

6.3.2 End-host Traffic Modeling

To answer those questions, an analysis of the Dartmouth College campus wireless traces was performed. The study was based on the set of 4-month traces collected in the *residential* buildings on campus [87]. First, the patterns of remote IP addresses [3] to which an individual user connected were studied. Figure 6.2(a) shows the individual remote IP addresses that a specific user contacted during one day. In the figure each circle represents a sanitized IP address. Obvious horizontal "lines" can be observed which implies that some specific IP addresses are frequently accessed by the user. To further quantify the temporal correlation of the remote IP addresses visited by the user the auto-correlation of IP addresses in Figure 6.2(b) is shown. It can be seen that the normalized correlation is above 0.85 even when the time lag is increased to

[3]Note that IP address sanitizing in Dartmouth traces is prefix preserving.

as large as 100 seconds.

The strong temporal correlation suggests that it is possible to make online predictions about the IP addresses a user is most likely to visit according to recent history. From this fact a simple prediction strategy can be designed. For each previously visited destination address a list of related IP addresses is maintained. After a destination is visited, the list of related addresses is updated with the connections that follow. The list is sorted based on frequency of visiting each address. With high probability, the user will connect to one of the first m IP addresses from the related list in the near future. This prediction strategy is preferred over more sophisticated models (e.g., Markov model) for its simplicity.

Next, the traffic volumes transferred between each particular remote host is studied. For each remote IP address, the traffic volume of all the connections directed to that IP address as a time series were extracted. It can be observed that the traffic exhibits burstiness over several orders of magnitude. This burstiness is a hindrance for volume prediction. The dramatic change of up to several orders of magnitude is hard to predict using traditional prediction methods. To address this challenge, a novel method to predict the order of the traffic volume using an on-line LMMSE predictor is used. The main power of prediction is not to keep track of accurate instantaneous value of the real result, but instead to characterize stochastic properties such as the mean value.

The flow volume series associated with a remote IP address for a typical user is defined as a time series $x(t)$ and define $y(t) = \log x(t)$. Analysis shows that the autocorrelation function of $y(t)$, denoted as $R(\tau) = \int_0^\infty y(u)y(u+\tau)du$ obeys the following rule:

$$R_y(\tau) \sim -a\tau + b, \tag{6.1}$$

where a and b are two constants such that $R_y(\tau) \geq 0$. The linearly decaying autocorrelation indicates that process $y(t)$ is strongly autocorrelated and therefore predictable with high accuracy. A LMMSE predictor is chosen for on-line estimation of $y(t)$, since it directly capitalizes on the auto-correlation structure and avoids the two-fold prediction errors of other model-based predictors such as FARIMA and FBM. The predictor is designed around the predictor used in predictive active queue management [61]. Specifically, given the time series $y(k), k = 1, ..., n$, the next series sample, $\hat{y}(n+1)$, is predicted in the next interval as a weighted sum of the past n history samples:

$$\hat{y}(n+1) = \begin{bmatrix} a_1 & a_2 & ... & a_n \end{bmatrix} \begin{bmatrix} y(1) \\ y(2) \\ ... \\ y(n) \end{bmatrix} \tag{6.2}$$

(6.3)

where $a_1, a_2, ..., a_n$ are the *LMMSE* coefficients and can be expressed as:

$$\begin{bmatrix} a_1 & ... & a_n \end{bmatrix} = \begin{bmatrix} R(n) & R(n-1) & ... & R(1) \end{bmatrix} \times \begin{bmatrix} R(0) & R(1) & ... & R(n-1) \\ R(1) & R(0) & ... & R(n-2) \\ ... & ... & ... & ... \\ R(n-1) & ... & ... & R(0) \end{bmatrix}^{-1} \tag{6.4}$$

where $R(n)$ is the covariance function of the time series. In practice this can be estimated as:

$$R(i) = \frac{1}{n} \sum_{t=i+1}^{n} y(t)y(t-i), \quad 0 \leq i \leq n-1 \tag{6.5}$$

where n is the number of the time series samples kept and is a tunable parameter.

Figure 6.3 shows the on-line predicted flow volume versus the real traffic volume over a period of 10 seconds using the Dartmouth traces to drive the predictor. The mean prediction error using the predictor is around 80% compared to a mean prediction error of 1000% when directly applying LMMSE to $x(t)$.

6.3.3 Hybrid Flow Scheduling

The analysis results obtained in Section 6.3.2 are used to guide the design of PERM's hybrid scheduler. On flow establishment phase a flow is classified based on its predicted volume from the LMMSE predictor. If the flow volume is light the flow is scheduled to the link with lowest RTT. The IP prediction model described above is used by PERM to pre-probe Internet destinations to measure the RTT on each available link. A threshold of 20KB is used to determine heavy-volume flows. In the analysis it was found that the median flow volume was 4383 bytes. Nearly 60% of the flows had volume under 20KB. For TCP flows,

20KB will typically be transmitted before the congestion window reaches only 10KB which should not be enough to reach the bandwidth constraint.

There are two possible approaches to selecting the best link for a heavy-volume flow. "Minimizing the total transmission time" is equivalent to minimizing the time to finish all existing backlogged downloading as a group, treating all flows equally. "Min-Max transmission time", on the other hand, minimizes the maximum download time of any one flow, favoring short flows with lighter volume. Assume the number of available access links is M, and each has an available bandwidth of $B_j, j \in \{1, 2, ..., M\}$. The number of flows being scheduled is N, and each has a predicted traffic volume $V_i, i \in \{1, 2, ..., N\}$. Let the scheduling matrix be C, an $N \times M$ array, and the element $c_{i,j} = 1$ denotes that flow i is scheduled to link j, otherwise $c_{i,j} = 0$.

It is assumed all the flows are TCP flows with pre-probed round trip times $RTT_i, i \in \{1, 2, ..., N\}$. Since in steady state TCP flows share the bottleneck bandwidth inversely proportional to their RTT^2 [108, 117], it can be obtained that for link B_j, the achievable bandwidth $A_{i,j}$ for flow i is $A_{i,j} = (B_j \cdot c_{i,j})/(Z \cdot RTT_i^2)$, where $Z = \sum_{i=1}^{N} \frac{c_{i,j}}{RTT_i^2}$. Given $A_{i,j}$ and V_i, the expected lifetime of flow i is therefore $V_i/A_{i,j}$.

The two scheduling decisions can be formulated as:

Minimizing total transmission time

$$\begin{array}{lll} \min & \sum_{i=1}^{N} \frac{V_i}{A_{i,j}} & \text{for } j \text{ such that } A_{i,j} > 0 \\ s.t. & c_{i,j} \in \{0, 1\} & . \end{array} \tag{6.6}$$

Min-Max transmission time

$$\begin{array}{lll} \min_{c_{i,j}} \max_i & \frac{V_i}{A_{i,j}} & \text{for } j \text{ such that } A_{i,j} > 0 \\ s.t. & c_{i,j} \in \{0, 1\} & . \end{array} \tag{6.7}$$

Both problems are integer non-linear programming problems and are NP-complete. Although the number of concurrent heavy-volume flows initiated by an end-host is unlikely to be big, N, C and $A_{i,j}$ are all functions of time as flows dynamically arrive and depart. The time-evolving N, C and $A_{i,j}$ make the above optimization problems prohibitively expensive to solve. Therefore, a heuristic-based greedy algorithm is proposed to ensure low scheduling delay.

Note that for the "min-max transmission time" scheduling discipline $\max_i \frac{V_i}{A_{i,j}} = (\sum_i^N V_i)/B_j$. It is therefore straightforward to add the volume of the new flow to the total remaining load on each link, and divide the total load by the link bandwidth. The new flow will then be scheduled to the link with the earliest finish time. The competitive ratio is at least $\frac{(3M)^{2/3}}{2}(1 + o(1))$ according to [29].

For the "minimizing total transmission time" scheduling discipline, it is necessary to calculate the finish time of the new flow efficiently. To this end, a sorted list of flows is maintained that are currently scheduled on a link, according to the flows' current finish time. Without loss of generality, a link 1 is picked with available bandwidth B_1, and it is assumed that before f_{new} comes, there are N_1 scheduled flows. Let $A_i, i \in \{1, 2, ..., N_1\}$ be the achievable bandwidth for flow i and $A_i = B_1 / \left(\sum_{j=1}^{N_1} \frac{1}{RTT_j^2} \cdot RTT_i^2\right)$. Since TCP flows share the bandwidth proportionally, adding new flows will not change the order of existing flows' finish time. Based on this observation the following theorem is derived:

Theorem 1. *Let V_i be the traffic volume of flow i, RTT_i be the current RTT, B is the available bandwidth for the link and N_a flows share the link including the new one. Then, the kth flow in the sorted list will finish transmission at time T_k:*

$$T_k = \frac{\sum_{i=1}^{k-1} V_i}{B} + \frac{\sum_{i=k}^{N_a} \frac{1}{RTT_i^2} \cdot V_k}{\frac{1}{RTT_k^2} B} \tag{6.8}$$

The list of flows is sorted in ascending order of $V_i \cdot RTT_i^2$.

Proof. Since flows share the link bandwidth proportionally, they finish in the order of $V_i \cdot RTT_i^2$. For those $k-1$ flows that end before the kth flow, their transmissions cause a total delay of $\frac{\sum_{i=1}^{k-1} V_i}{B}$ for the kth flow. For the remaining $N_a - k + 1$ flows, they share the link bandwidth proportionally until the kth flow ends, which brings another delay for the kth flow as the second item in Eq. 6.8 indicates. □

Given Eq. 6.8 the scheduler explicitly calculates the total increase in flow finish time at each link, and schedules the flow to the link with minimum increase.

6.4 Implementation

The PERM client software has been implemented for both Linux and Windows. As illustrated in Figure 6.4, the implementation of PERM is composed of two software packages. The access point software runs the privacy filtering and incentive management with traffic shaping. On each PERM client, the flow scheduling is implemented as a combination of shim library[4] to intercept system function calls from individual applications and a scheduling daemon for all applications. Users may elect to run PERM without traffic shaping in which case no software installation is needed on the access point. In total, the implementation has around 12K lines of C code.

[4] A small library that converts one API into another.

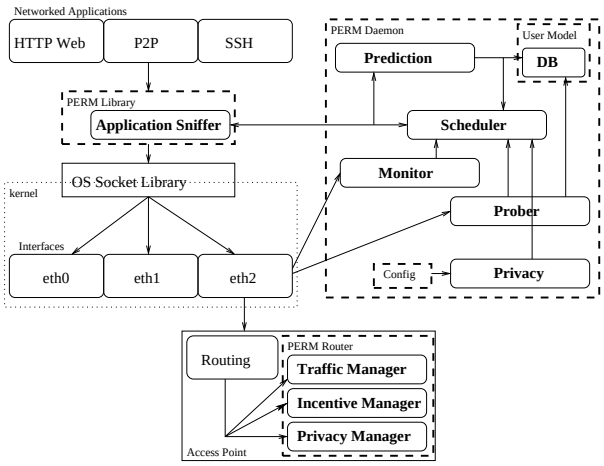


Figure 6.4: The PERM Framework for the Linux implementation.

6.4.1 Certified Identity Control

As a general requirement for both traffic shaping and security a user must be able to identify which clients and gateway belong to the same owner. The PERM system uses AGE authentication as described in Chapter 7. AGE uses digital certificates to bind all of an entity's devices to the same unique name, called the `certified name`. Each PERM owner has a single certificate that is installed on the owner's gateway and all of the owner's client devices, thus identifying those devices by the same certified name. Mutual authentication is performed where both the access point and client exchange and validate the other's certificates. During this process the certified name is extracted from the certificate and stored for future identification.

6.4.2 PERM Wireless Router

Upon successful verification of the client certificate the identity credentials are passed to the Perm Client Manager that installs the necessary filters and routing rules for the new client. A series of `iptables` filters are installed to sort traffic into the correct class. Since WPA is a link-layer security mechanism, it does not prevent an authenticated user from changing their IP addresses. Therefore clients are identified using their MAC address in the `iptables`. If the access point and the client share the same certified name, the client is considered a local client. Otherwise, the client is considered foreign. When the owner is inactive, he might choose to share with his neighbors at a higher rate so, in the owner idle state, foreign traffic is assigned to the high class. If the owner is active, credited foreign traffic is always assigned to the lowest class. To enforce the traffic shaping, the HFSC (hierarchical fair service curve) [125] class-based traffic scheduler that comes with the Linux 2.6 kernel is applied. The PERM Client Manager is implemented in the open-source Linksys WRT54G, a popular, highly-customizable home wireless router. It has four

Function	Linux	with PERM	Increase (%)
`socket`	0.203 usec	2.294 usec	2.091 usec (1030.0%)
`setsockopt`	0.047 usec	0.054 usec	0.008 usec (16.1%)
`getsockopt`	0.047 usec	0.051 usec	0.004 usec (8.6%)
`bind`	5.62 usec	84.45 usec	78.83 usec (1402.7%)
`connect`	15.38 usec	282.25 usec	266.87 usec (1734.8%)
`send`	4.159 usec	4.925 usec	.766 usec (18.4%)
`recv`	18.51 usec	0.667 usec	-17.84 usec (-96.4%)
`close`	1045.9 usec	836.57 usec	-209.34 usec (-20.0%)

Table 6.1: Average elapsed time, in microseconds, of network system calls. Times of the normal Linux system calls are listed on the left. Times of the calls in the PERM framework are in the middle. The increase is given in the final column.

10/100 Ethernet ports, one WAN port and one 802.11g wireless interface. The version used is equipped with 16MB RAM, 4MB flash and 200MHz MIPS CPU. The router runs the OpenWrt Linux [11] operating system, one of many different Linux distributions for WRT54G developed by the open-source community.

In order to determine the real effect of the increased latency, loading a web page with PERM scheduling and with load balancing scheduling were compared. The average time for `wget` to fetch the index page of a popular web site was measured 100 times. Despite the relatively higher computational overhead, PERM adds only 0.04 second increase to the total transmission time. The load balancing scheduler adds comparatively less overhead, only 0.01 second in total. The overhead for the sliding window scheme was the same as that for PERM scheduling. Therefore, despite the computational overhead PERM is still able to achieve considerable performance improvement even for light-volume flows.

6.4.3 PERM Library

PERM scheduling is implemented by two tightly coupled modules, the PERM library and the PERM daemon. Using `LD_PRELOAD`, a Linux environment variable which specifies libraries to load before applications are executed, the PERM library intercepts application calls to create new network connections. When a new connection is detected, the library contacts the daemon that tells it the IP address of the local network interface onto which the connection should be bound. The library then `bind`s the socket to the given interface. The Linux routing tables are modified to route packets between the different interfaces based on source address. After transferring 10KB or when a connection is closed, the library reports the bytes transferred to the daemon which then sends a credit report to the local access point. The library also exports an extended socket API to allow for programmed interaction with applications. This allows for applications to force connections to specific interfaces and specify flow metrics such as expected flow volume.

By modifying the original `libc` socket library calls, some extra processing overhead is introduced. Using a Pentium III workstation the overhead increase for each of the intercepted function calls was measured. A client connected to a server over the `127.0.0.1` local interface and read a short message. Over

1000 iterations were performed. Table 6.1 shows the average times as measured in microseconds. For most of the functions, the change is negligible. For the `connect`, `bind` and `socket` calls, there is a large increase. The primary overhead in each of these functions is the added IPC message (in the case of `connect`, two messages) from the library to the daemon. Although these calls have an increased overhead, it is a one time cost for each connection. The per-packet overhead is negligible and has no noticeable effect on overall performance as shown in the traffic evaluations. It was surprising to discover that the average times for the `recv` and `close` call actually *decreased* using the PERM library. For the `recv` call it is likely that some of the time that the default implementation spends waiting for the next available packet is spent in different parts of the PERM library not captured in the `recv` call.

The PERM library also exports an extended socket API by which an application can provide specific flow information. It is important to stress that because of predictive scheduling, an application can receive the benefits of the PERM framework without using this special API. However, scheduling can be improved with the extra information that an application might provide. Flow attributes are set via the `setsockopt` call. The PERM library defines a new socket level `SOL_PERM`, which distinguishes PERM options from other socket options. If the socket level in the `setsockopt` call is set to `SOL_PERM`, then PERM processes the options. Otherwise, the call is forwarded to `libc`. Currently, PERM allows an application to specify four flow characteristics: volume, delay, jitter, and priority. The first, volume, is the number of bytes the application plans to transmit/receive on the connection. The other three, delay, jitter, and priority are requirements that the application desires to have. Each of the latter attributes can be given a value of either HIGH, MEDIUM, or LOW, as well as hard values.

6.4.4 PERM Daemon

The PERM daemon runs continuously in the user space on the client system, accepting connections from the PERM library via a UNIX local socket. The daemon contains the prediction, probing, monitoring, privacy and scheduling components. To monitor each access link, profiles for each available network interface are maintained. The profile contains the IP address, certified name of the gateway router, current flows using that interface and estimated capacity of the link. To implement the prediction techniques from Section 6.3.3 the daemon maintains an IP database containing previously visited IP addresses and for each a cache of likely related IP addresses. The RTT to a given IP address is probed on each interface using TCP `SYN,ACK` packets [24]. To measure round trip time the daemon probes each of the hosts in the target list every T seconds, set to 60 in the implementation. The probing is done by sending a series of `SYN,ACK` packets to the target host. The `SYN,ACK` packets cause a `RST` packet to be sent from the remote

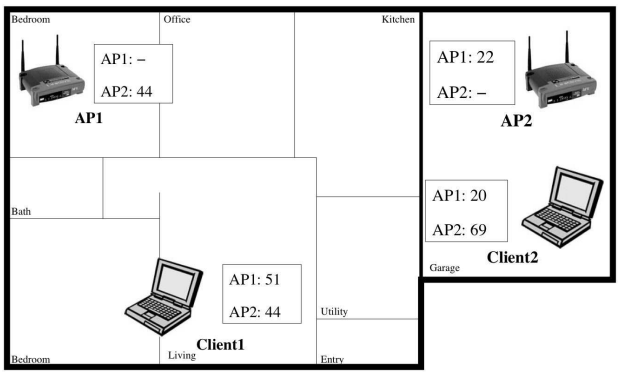


Figure 6.5: Experimental Deployment. Numbers next to each device represent the average signal strength to each of the access points.

host which is used to measure the RTT. A probe is retransmitted three times with an exponential timeout before a host is declared unreachable.

6.5 Performance Evaluation

The PERM implementation was deployed on a testbed in one author's home. The testbed consists of two Centrino duo laptops running the 2.6 Linux kernel, and two LinkSys WRT54G wireless routers. Each laptop is equipped with an internal Intel 802.11a/b/g wireless interface, as well as an external NetGear 802.11g USB card. The two wireless routers are connected to two different ISPs, one a DSL and the other a Cable provider. The maximum achieved throughput during the evaluation on the Cable connection was 5.5Mbps downlink and 900Kbps uplink. For the DSL line the speeds were 1.32Mbps downlink and 400Kbps uplink. Figure 6.5 shows the layout of the testbed within the house. The numbers next to each device show the average signal strengths to each access point. In order to best emulate having access points in different homes, one access point is placed in the garage and the other in the house. This deployment accurately represents the situation in many apartments and closely mimics sharing between houses. For example, the average signal strengths from three different access points in surrounding homes was 29, 21 and 14 while the average signal strength from Client2 to AP1 was 20 and from Client1 to AP2 was 44.

6.5.1 PERM vs. Other Schedulers

First, the performance of PERM's hybrid scheduler was compared against several existing multihoming schedulers from the literature. To this end, both hash-based and load-balancing scheduling algorithms [63] were implemented as well as sliding window active probing [24]. Hashing and load balancing are simple, light-weight approaches which do not use any information about the underlying links. To improve schedul-

Scheduler	Mean Transmission Time in sec.	
	Light Volume	Heavy Volume
PERM (Min-Total)	1.50487	186.21
PERM (Min-Max)	-	199.14
Active Probing	1.71785	201.23
Load Balancing	2.04534	211.37
Hashing	2.07801	257.57

Table 6.2: Mean transmission times for light- and heavy- volume flows under different schedulers during controlled experiments.

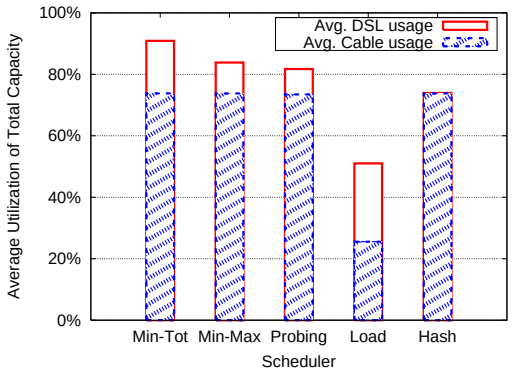
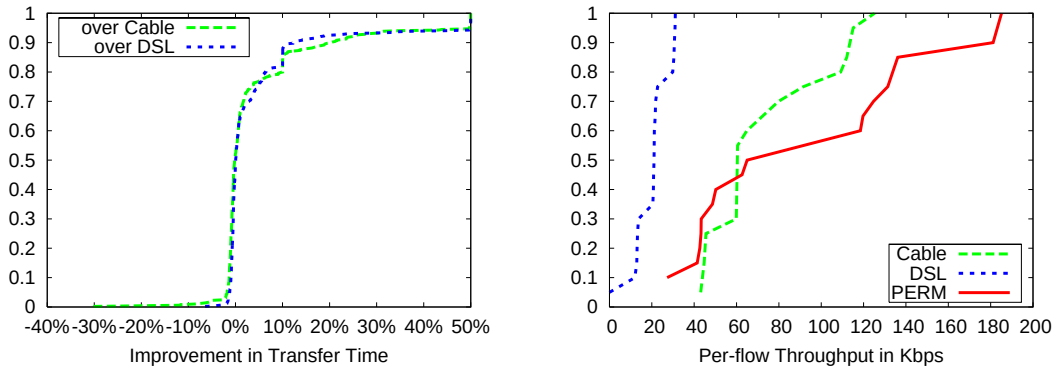


Figure 6.6: Average utilization of total available capacity from both access links combined during heavy-volume controlled experiments.

ing performance, the links can be probed to profile link characteristics allowing the scheduler to make more accurate decisions. The results confirm that there is a performance improvement of probing-based approaches over simpler approaches. The experiments further demonstrate that PERM's predictive mechanisms lead to higher performance than other probing-based schedulers.

Light-volume Flows

To quantify the benefits coming from PERM's more accurate probing, the transfer times of small web connections were measured with different multihoming schedulers. The experiment was run using traces derived from several of the busiest users in the Dartmouth traces not used in the traffic analysis from Section 6.3.2. Because IP addresses in the original traces are sanitized, the IP addresses in the traces were replaced with the IP addresses of the top 500 web-sites as listed in the end-user based web-site rankings from the Alexa.com web information archive. The original access pattern from the traces was unchanged. For each scheduler the index page from each web site in the trace was downloaded. Multiple iterations were conducted for each scheduler at different times of the day. The first column of Table 6.2 shows the overall mean transmission time for each scheduler. The PERM scheduler has the lowest overall mean trans-



(a) CDF of Improvement in Light-Volume Flow-Transmission Times

(b) CDF of Heavy-Volume Per-Flow Throughputs

Figure 6.7: Controlled Experiments: (a) Percent improvement in transmission times for light-volume flows. (b) Per-flow throughput for bulk transfers.

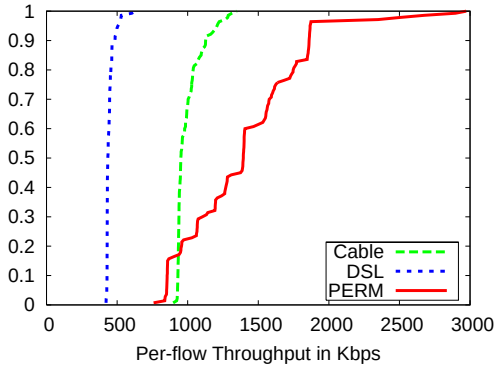


Figure 6.8: CDF of Per-flow throughput for upload flows.

mission time, 38%, 35%, and 14% lower than the other schedulers. Considering that light volume flows take up more that 60% of all connections, these results confirm that there is noticeable benefit from active probing in general and even more so from PERM's predictive probing.

Heavy-volume Flows

The transmission time of heavy-volume flows is constrained more by available bandwidth rather than RTT. Because small flows do not compete for bandwidth with the large flows, to evaluate the effectiveness of the PERM scheduler for managing the available bandwidth several large files from the Internet were downloaded simultaneously. In the Dartmouth traces, 50% of users averaged only 1.25 or fewer simultaneous large-volume flows of more than 20KB. 96% of users averaged 4 or fewer simultaneous flows. Of the large-volume flows, 92% were less than 1MB although transfers of over 2GB were also present. Files of slightly larger size are used for the experiment to ensure that downloads would sufficiently compete with

one another and stress the system. Files were hosted on different web-servers ranging in size from 1.7MB to 30MB covering 99% of all transfers in the traces.

To test the schedulers, several different iterations of downloading 4 different files simultaneously were performed. The downloads combined saturated the underlying links. The two PERM heavy-volume schedulers described in Section 6.3.3 are compared against the other multihoming algorithms. The mean total transfer times over 15 experimental iterations from each scheduler are shown in the second column of Table 6.2. The two PERM schedulers outperform all other schedulers although the Min-Max scheduler was only slightly better than the Adaptive Probing scheduler. The Min-Total scheduler improves the overall download times by 27.7%, 11.9% and 7.4% over each of the non-PERM schedulers. More insight into the relative performance of each scheduler can be gained by examining the utilization of each access link. Figure 6.6 plots the average utilization of the total combined bandwidth available during the duration of the experiment. The usage of each individual link is shown as two separate parts of the total utilization. Again, the two PERM schedulers were able to make most use of the total capacity. Moreover, the Cable with its larger bandwidth is favored by both PERM schedulers. Although the DSL was under-utilized (38% for Min-Max and 65% for Min-Total scheduler), the utilization of the combined total capacity is highest.

6.5.2 PERM vs. Single Access Link

Given PERM's effective scheduling techniques, next the possible benefits are quantified when using PERM compared to a single access link. As an aide in the evaluation, the top 150 web sites as listed at Alexa.com were repeatedly probed using the probe method described in Section 6.4 to capture RTT characteristics and link profiles. The number of hosts probed was limited so that every host could be probed at least once every 5 minutes. Probes were initiated simultaneously on both links to ensure that Internet-wide conditions were similar. Each remote host was probed on each interface around 2000 times over a period of one week. Hosts that did not respond to any probe were removed from the target list.

As a baseline for comparison, the benefit a scheduler with perfect knowledge can gain is first establish. Using a model for duration of short-lived TCP flows [100] and the week-long RTT measurements, it was found that a scheduler which always chooses the lowest RTT link could achieve 11.7% improvement in transfer times of light-volume flows over Cable and only 2.9% improvement over DSL. Although these results are very particular to the conditions measured during the one-week study and used only a subset of the hosts used in later experiments, they provide an expectation to gauge further results. The same controlled experiments as in Section 6.5.1 were next perfomed to see how PERM's scheduler compares to sin-

gle link cases for web traffic, bulk downloads, and file uploads. Finally, the potential reliability gains under PERM were analyzed.

Light-volume Flows

To evaluate the performance for web surfing, the user traces were used to schedule the download of web index pages. Each index page was downloaded and the length of the entire transfer was recorded for each instance and compared.

The flows for which the download failed for every request on all interfaces were removed. 9 connection attempts failed for every attempt on Cable but succeeded with PERM while 11 connections failed on DSL but succeeded with PERM out of 450 total. The average per-host transfer time over each access method was computed. Ideally, PERM's prediction mechanisms would always lead to selecting the interface with lowest round trip times and thus lowest transfer times. Given that the above RTT measurements indicated that a given access link was about 50% likely to have the lowest RTT to a given destination address, it would be expected that PERM would out-perform each individual access link for about 50% of the hosts. Indeed, the experimental results show that for 46% of the flows PERM out performed the Cable and for 45% of the flows PERM out performed the DSL. For an additional 28%/38% of the flows, PERM download times were within ±1%. For the rest, the PERM scheduler performed worse than the other access method. This means for about 17-25% of the flows, conditions on the access links changed before PERM could detect it, or PERM mispredicted and scheduled to the wrong interface.

Figure 6.7(a) summarizes the distribution of per-flow percent improvement in mean transfer times using PERM compared to the two single access links. Comparing only the flows that succeeded in both PERM and on the single access link PERM achieved a 11.8% improvement over Cable but less than 1% improvement over DSL. These results are consistent with the results of the perfect scheduler above.

Heavy-volume Flows

Using the bulk-flow experiment from Section 6.5.1, PERM's performance in downloading large files was compared to single links. The per-flow throughput achieved in each scenario is measured. Although the peak flow throughput is limited by the underlying access links, because of PERM's effective distribution of flows, the average per-flow throughput is higher. The cdf of per-flow throughputs is shown in Figure 6.7(b). The average flow completion time for PERM was 28% lower than Cable and 62% lower than DSL. Note that although PERM does not outperform single link cases for every individual flow, the aggregate transfer time is minimized. Considering the 4:1 downlink capacity ratio between Cable and DSL in the testbed,

Access	Mean Resp. Time	Failure Rate
Cable	56.431 ms	1 in 10.06
DSL	55.138 ms	1 in 14.96
PERM	51.823 ms	1 in 16.33

Table 6.3: Mean performance metrics during longitudinal study for PERM compared to Cable and DSL access links.

the performance gain of PERM is close to optimal.

Uploads

Because of the popularity of p2p applications and swarming downloads, the utilization of the access link is becoming much more symmetric than before. PERM can schedule outgoing flows by controlling the interface on which an application binds it's socket. Without specific application support, PERM can only schedule outgoing flows on the application level as individual flows initiated by remote hosts are beyond the control of the end-host. To measure upload throughputs, two file servers were started on the PERM host. Multiple download requests were initiated from different remote hosts in a campus lab. Figure 6.8 shows the distribution of per-flow throughputs for the uploads. Clearly PERM has large gains in throughput even though the DSL upload speed was around half that of the Cable link. Even when scheduling flows to the DSL link only 15% of the PERM flows had lower throughput than Cable flows.

Reliability

Finally, the failure characteristics of the Cable and DSL links is quantified and the potential improvement in resiliency that might be possible with PERM scheduling analyzed. To measure the overall failure rate, the above week-long RTT measurements are used. A connection failure is defined as any probe attempt that does not receive a response within a 3 minute time period. After a failure, probes were not retransmitted.

The total failure rate of the probes was 20.3% on Cable and 12.8% on DSL i.e., 88,994 and 56,114 probes out of the total 438,397 on each link failed. Of those failures only 3292 (4% of Cable, 6% of DSL) were simultaneous on both links, meaning that around 94% of the failures could potentially be avoided with PERM scheduling. These failure rates measured by probing might be high due to temporary packet loss or because some TCP/IP implementations ignore the `SYN,ACK` packet. In the longitudinal deployment study presented in the next section, the real world reliability improvement is verified.

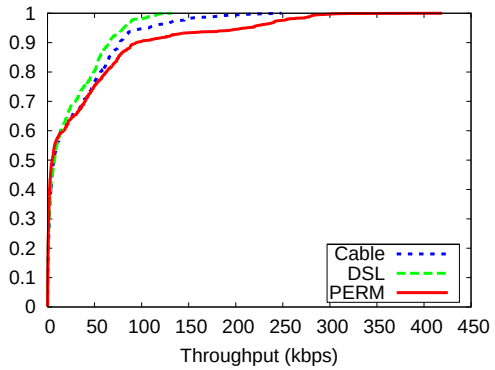


Figure 6.9: CDF of throughputs achieved for heavy-volume flows during longitudinal deployment.

6.5.3 Deployment Results

After running the controlled experiments with trace driven generated traffic, a longitudinal deployment study was performed with two users running PERM during their normal day to day Internet usage. `tcpdump` was used to collect a packet capture of each user's traffic. Traces were captured under three scenarios, running only on Cable, only on DSL, and with the complete PERM system. The traces in total spanned a five week period with nearly 2 million packets. The majority of flows were HTTP traffic but there was also significant IMAP, streaming MP3, and ssh traffic. There were few heavy-volume flows. Although these behaviors may not represent all ranges of Internet activity, the results apply to many residential scenarios.

Light-volume Flows

Based on the packet capture, there was an observed 26.1% and 14.0% increase in per-flow throughput for small-volume flows under 20KB. Because the longitudinal study was not controlled, the set of remote hosts for light-volume transfers was not the same for all access methods meaning the transfer time of flows is affected not only by the scheduler but by the different sets of transfer sizes. Another metric that indicates the perceived performance improvement is the per-flow average response time. The average response time is measured as the time between the TCP `SYN` being sent and the first TCP `SYN-ACK` returning. The first column of Table 6.3 shows the average response time for web connections under each access method after removing the outliers. PERM's mean response time is 6.0% and 8.2% lower than DSL and Cable. Because the latency is reduced for every packet there is higher perceived performance for the end user.

Heavy-volume Flows

In the traces, there were far fewer heavy volume flows than light volume flows - about a 1 to 10 ratio. Over the course of the deployment, the largest file that any user transferred was 23 MB, but most flows were smaller than 1 MB. Also the number of simultaneous transfers was lower than in the controlled experiments which means there was a lower link utilization and therefore less opportunity for the PERM scheduler to maximize benefit. Figure 6.9 shows that there is still an overall per-flow throughput improvement while running PERM even under low stress.

Reliability

One of PERM's most attractive enhancements is avoiding failures in the access network. The traces were examined to find the number of failed connections when running in each access method and compare to the estimated results of Section 6.5.2. In this case, a failure is defined as any TCP connection attempt that times out before completing. It was found that although the failure rate in the traces was lower than Section 6.5.2, the Cable link had a failure rate 62% higher than PERM. The DSL suffered a nearly 10% higher rate than PERM. The results are summarized in the second column of Table 6.3. These reliability improvements came during normal Internet conditions with no major outages detected. In the face of major ISP failure PERM would be even more beneficial.

6.6 Future Work

In the description of PERM's design, it was assumed that the software was deployed on end-hosts. While this has advantages for scheduling because the traffic is generated at end-hosts, the approach creates a roadblock for PERM's deployment since PERM must be installed separately on all clients to be used. Ideally, the scheduling framework would be moved from the clients into the network itself. Stable nodes could form dynamic mesh networks around the existing gateways. The mesh nodes would forward all traffic from the clients and load-balance it according to the mechanisms described in this work. There are several challenges to this approach including gateway promotion, mesh stability, and distributed traffic scheduling. The Coalition-Based Peering [90] and COMBINE [26] projects might offer a starting point for designing a system for the UCN.

6.7 Conclusion

Widespread deployment of home wireless networks in residential areas enables ubiquitous Internet connectivity. The PERM framework harnesses this connectivity to solve residential broadband performance shortcomings at low cost. The PERM system is practical, not relying on any support outside of the target home. Further, PERM uses predictive mechanisms to achieve high-performance flow scheduling. PERM also preserves a user's security and privacy and enforces incentive management to encourage the participation of compliant users. Through both controlled experiments and live deployments, it was shown that PERM's hybrid scheduler achieves significant performance gain over existing enterprise multihoming schedulers. Also, the PERM framework clearly offers considerable throughput and latency improvements compared to using a single Internet connection.

Chapter 7

Authentication for Managing Inter-network Communication

Authentication is needed to manage shared gateways in the UCN Internet access model. Using existing authentication systems in the UCN will result in long authentication delays and unreliability. Because there is no central operator in the UCN, each gateway potentially connects to a different ISP. Existing authentication systems require deploying servers in the Internet where they are vulnerable to high and unexpected delays, loss of connectivity and attack. In this chapter, a distributed, local authentication framework is developed. By constraining authentication to local interactions only, low-delay and high-reliability authentication is enabled for the UCN. To further reduce delay from high computation overhead, authentication material is segmented using a location-based mechanism. Deployment in PlanetLab and a residential testbed shows the reduced delays.

7.1 Global Authentication in User-created Networks

ISPs will only allow devices from the UCN to access their network if there is strict access control to ensure traceability and accountability, required for legal and regulatory reasons by the ISPs. Leaving the management of such a system to individual end-users, as many community projects do (see [34] for a summary), is not viable because most end-users are not willing or technically able to handle such tasks. Authentication and access control overseen by a central authority is a must. Given the global nature of the user-created network, individual ISPs are ill-positioned to act as central authorities. What is needed is a global authentication service run by a trusted third party.

The difficulty in deploying a global authentication service comes from the nature of the UCN. While a third-party authentication service is desirable for the flexibility it provides UCN clients, it means any supporting servers and infrastructure must be deployed in the Internet, rather than in the backhaul networks. This fact, combined with the unique conditions in the UCN, mean that existing authentication mechanisms are unfit in UCNs for four main reasons. First, existing authentication methods are not designed to scale to the size of the UCN. Intended for deployment in enterprise LANs with hundreds or maybe thou-

sands of clients the authentication server will be seriously challenged with potentially millions of active clients.

Second, the UCN system acts as an overlay operator (or an overlay wireless ISP) on an access infrastructure that is collectively owned by its actual users. Network management systems currently used in enterprise access network deployments [131] are not suitable in this context due to the lack of control over the interconnection medium in the UCN. Centralized authentication is vulnerable to general Internet outages, loss of connectivity in the UCN and distributed DoS attacks. The access points should be able to operate reliably and autonomously, even when access to network functions provided by the back-end management system is temporarily unavailable or inaccessible. Ideally, nodes in the UCN could authenticate each other without any access to the infrastructure.

Third, to deploy existing authentication systems in the UCN requires deploying authentication servers in the Internet, spacing the authentication server far away from both clients and access points. Network anomalies, latency, and intermittent loss of connectivity are frequently encountered in wide area networks at the Internet scale. Authentication in the user-created network must be fast because of the limited connection opportunities and node mobility.

Finally, as access points in the UCN multiplex wireless connectivity of their owners and other nomadic visitors, multiple wireless networks with different security profiles, traffic shaping and prioritization policies need to be advertised and served by each of these access points. Virtualization of the access points [17] is a key requirement towards the wide acceptance of UCN among performance and security concerned users. UCN should provide equivalent wireless security provisioning for both the public and the private networks.

This chapter presents the design and implementation of a scalable and resilient service for *authentication on the edge* (AGE) of UCNs that is used for access control of the nomadic users. A set of *semi-distributed* algorithms and protocols operate under light-weight centralized coordination to authenticate users for controlled access. The design of AGE seeks to strike a balance between fully centralized approaches and fully distributed approaches in order to capitalize on the simplicity and ease of management of the former and scalability and ideal robustness of the latter. AGE's mechanisms allow the bottleneck and single point of failure found in existing authentication schemes to be avoided. AGE's mechanisms make it well suited to the UCN. AGE supports a single authentication authority, allowing clients to access the service anywhere in the world with the same user id and authentication credentials. Authentication in AGE proceeds with as little user interaction as possible (the user only has to select the UCN SSID for association) and AGE is resilient to the variable network conditions in UCNs including potential loss of connectivity to the cen-

tral server.

At its foundation AGE is based on EAP-TLS authentication [16], an EAP method using certificates and private keys. The TLS authentication method avoids the use of passwords and allows mutual authentication of the authenticator (access point) and supplicant (mobile client). A central server operates the AGE certificate authority (CA) which manages the certificates for all users. To overcome the unreliable connection to the central server, each AGE access point (AP) runs a self-contained authenticator, confining the authentication process to the wireless link only. In addition to its certificate, each device also carries the most recent CRL proving the certificate has not been revoked. To manage the limited storage and computation resources which might not be able to handle a large CRL for a global network, the CRL is segmented to reduce resource demand. Without access to an authoritative server, there is a trade-off in the opportunistic use of whatever authentication material is available and the certified freshness of the available material. If the APs freshness demands require tighter guarantees then the available material allows, p2p look-up mechanisms can be used to verify that the presented CRL segment is indeed the newest one.

Through evaluation it is shown that AGE is immune to Internet delays and achieves low authentication times and correct operation in the face of failure of the central server. Measurement results comparing EAP-AGE to EAP-TLS in the UCN scenario show that AGE satisfies requests with between 49.7% and 71.6% lower delay (around 490 msec. and 1614 msec.), providing a faster and more predictable authentication. Using SMS traces from a large nationwide cellular provider it is also shown that with only 5 social connections as routing paths, AGE's social network will provide a nearly 100% success rate of CRL segment lookups.

7.2 The Need for Decentralized Authentication

Existing authentication mechanisms are unfit in UCN because of the deployment and communication challenges in the UCN. There are two predominant approaches currently in use to manage access to wireless networks: captive portals and LAN port authentication schemes. Captive ports are deployed in hotspots and several open Wi-Fi projects [1, 2, 7, 14]. A captive portal is a firewall application which blocks all traffic going from the client to the Internet until the user has authenticated through a custom web page[1]. Once the user attempts to make a web connection her browser is directed to the authentication page where she enters her access credentials. The request is forwarded to a central database, usually through the Internet, which confirms or denies the request. Each time users connect to a new access point, they must

[1]see NoCatAuth (http://nocat.net/), ChilliSpot (http://www.chillispot.org/), WifiDog (http://dev.wifidog.org/)

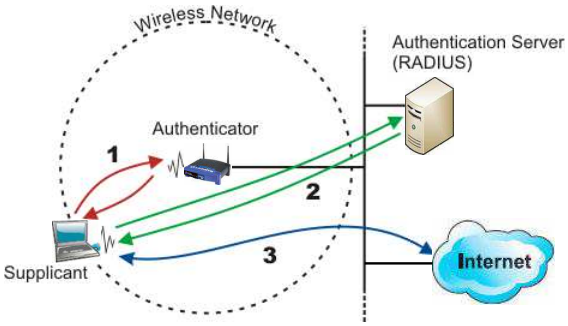


Figure 7.1: IEEE 802.1x LAN port authentication framework. The client (1) associates to an access point through which (2) it is tunneled to an authentication server on the LAN which finally (3) grants permission for outgoing connections.

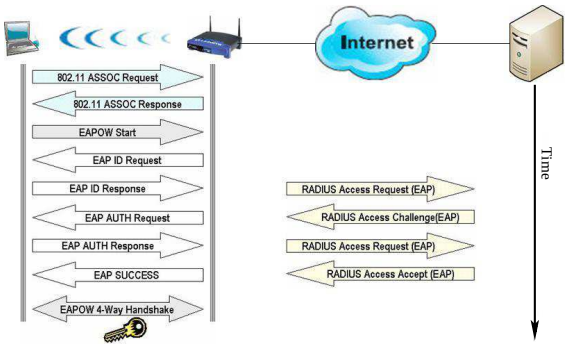


Figure 7.2: Direct application of IEEE 802.1x and EAP in UCN. The authentication server is located in the Internet requiring RADIUS tunneled EAP messages to cross the WAN.

re-authenticate with the central server. Authentication is done at the application layer so there is no possibility for layer 2 wireless encryption. Because captive portals are password based, provide no wireless encryption and suffer long authentication delays when communicating to the central server they are ill-suited for UCN networks.

Existing LAN port authentication schemes, such as WPA and 802.11i standards, are based on the IEEE 802.1x framework and the extensible authentication protocol (EAP) as illustrated in Figure 7.1. These schemes provide wireless encryption and support a wide range of different authentication methods. A supplicant, the client side access control entity, associates to an authenticator located in a LAN access point such as an Ethernet switch or Wi-Fi access point. The supplicant is authenticated through a closed port on the access point against an authentication server, for example RADIUS or MS Active Directory server. If the authentication is successful the authentication server signals the authenticator to open the port to allow authorized traffic from the supplicant, e.g. access to servers in the LAN or Internet. Although the standardized EAP methods work well in enterprise networks, the user-created network changes many assumptions about the operating environment which reduce the effectiveness of existing methods (as illustrated in Figure 7.2).

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Failure Rate	.8%	1.1%	2.1%	1.4%	1.2%	.1%	0%

Table 7.1: Per day failure rates of EAP-TLS authentication attempts over a cross continental link over a one week sample period.

The unreliable wide-area connection has considerable impact on the existing IEEE 802.1x authentication process. The entire authentication process using EAP methods involves several rounds of communication between the supplicant and authentication server. Tunneling EAP messages through RADIUS does not use reliable transport meaning it is vulnerable to packet loss and high and variable delay in the completion time. In the case of packet loss, it becomes difficult to gauge an appropriate timeout before restarting the authentication.

To quantify the effect on authentication success rates a measurement study of WPA EAP-TLS authentications in a UCN-like deployment was performed. A RADIUS authentication server configured for EAP-TLS authentication was installed on the UIUC campus in Illinois, USA. A client supplicant at Deutsche Telekom Laboratories in Berlin, Germany attempted an EAP-TLS authentication through a local wireless access point once every minute. Over a seven day period on average approximately 1% of all authentications experienced a timeout. A maximum of 2.1% of one day's authentication attempts failed. Compared to local deployments with close to 100% success rate the penalty is high. Note that the impact of every lost packet is considerable given the default 30-second timeout in most implementations. Note also that these measurements were also taken under favorable conditions: the authenticator contacted an unloaded authentication server over a high speed LAN located at T-Labs in Berlin, over the Abilene Internet2 network to UIUC campus. Real deployments of AGE in residential areas connect over considerably slower and less reliable DSL or Cable connections. The capacity of the server and the network that hosts the server will be seriously challenged when handling the large-volume of authentication requests from the entire UCN. Also the impact of server failure will be much more severe when serving requests from millions of users.

Delays will be further compounded by the centralized design of existing schemes. The capacity of the server and the network that hosts the server will be seriously challenged when handling the large-volume of authentication requests from the entire UCN network. Also the impact of server failure will be much more severe when serving requests from millions of users.

Although the overloading problem can be alleviated by re-directing the request to other authorized delegates, it will significantly increase the operation and maintenance cost, requiring constant adaption of the backend infrastructure to respond to the UCN's unplanned growth. For example, the OBAN [50]

project focuses on improving inter-AP handoff in open Wi-Fi networks by dividing the community into cells similar to cellular phone networks. The cell manager serves as an authentication proxy for each cell. OBAN requires the placement of servers in every cell, or one per 50,000 users as specified by the designers. Given UCN's targeted global deployment the OBAN approach requires high maintenance overhead and might struggle to find an optimal deployment of manager servers to cover the user-created network. In addition, an OBAN-like approach in the UCN would require the cooperation of all ISPs involved to deploy servers in their networks. Similarly, OCSP [104] reduces central server load through the use of a status validation protocol. However, OCSP has many of the same infrastructure and deployment problems as other approaches [45].

Other research projects have studied mechanisms to provide trust in decentralized settings. The MoB project [40] provides an eBay like trading mechanism for sharing wireless access between individual users in public areas. MoB focuses on forming trustworthy relationships between unknown identities and accurate reporting of usage. MoB does not try to provide a static network and therefore does not address authentication and access control. P2PWNC [51] similarly focuses on ensuring fair trading rather than authentication in an open Wi-Fi network by using centralized servers to establish paths of trust for exchange of bandwidth credit. Both of these schemes could benefit from the accountability provisioned by AGE.

7.3 Authentication on the Edge

To address the challenges of scale, outages and delay *Authentication on the Edge* (AGE) for user-created networks is proposed. AGE localizes the authentication process on the access points and avoids the unpredictability and potential unreliability of Internet-based authentication. AGE is comprised of three components: (1) mutual authentication between an access point and a client using X509 certificates, (2) distribution of different certificate revocation list (CRL) segments to participants at different locations and (3) fallback p2p lookup of the CRL freshness. AGE achieves scalability and resiliency by excluding the on-line involvement of the back-end authentication/directory server as illustrated in Figure 7.3. AGE supports a single authentication authority allowing clients to access the service anywhere in the world with the same user id and authentication credentials. Authentication in AGE proceeds with as little user interaction as possible (the user only has to select the appropriate SSID for association) and AGE is resilient to the variable network conditions in the UCN including potential loss of connectivity to the central server and any other infrastructure.

The design of authentication on the edge (AGE) seeks to strike the optimal balance between fully

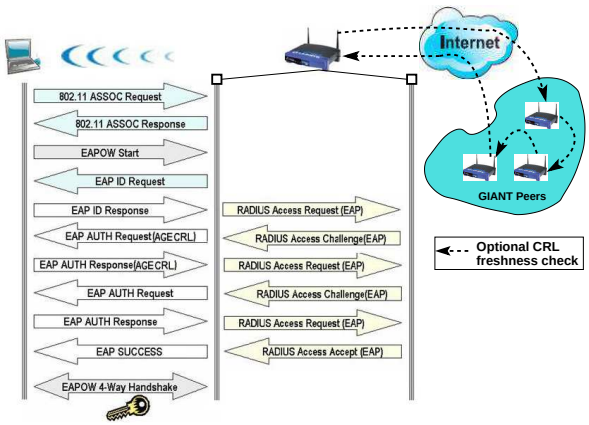


Figure 7.3: AGE authentication is fully localized between the access point and the client. *On-line* CRL freshness lookups are optional and rarely necessary.

centralized approaches and fully distributed approaches in order to capitalize on the simplicity and ease of management of the former and scalability and ideal robustness of the latter. Previous projects have shown that by distributing authentication information offline and allowing local authentication decisions, the authentication process itself is faster and more reliable but it comes at the expense of some freshness in user credentials. The same tradeoff has been exploited before for distributing certificate revocation lists [84, 101]. A higher off-line computation is invested at the directory server to speed up on-line lookup for revoked certificates. This tradeoff places a computation and communication cost on the directory server to generate and publish authentication material, but reduces the on-line communication overhead. AGE moves one step further by pushing the CRL database away from the directory server towards the Internet edge where the authentications actually take place, therefore saving not only on-line computation at centralized servers but also on-line communications with those central authorities over the Internet.

The AGE network is composed of two types of devices, access points (APs) and wireless clients. Typically each end-user owns only a single AP but might own many clients. Certificates are issued by a trusted central authority (CA) and are verifiable given the CA's public key. By installing the CA public key on each device, access to the CA is not necessary for most authentications. Certificate-based authentication also has the advantage that it requires no user interaction during authentication as compared to password based schemes.

During a certificate's lifetime, which is usually on the order of months or even years, it may be necessary to revoke the certificate because of key compromise, cancellation of service or for punitive reasons. The central authority maintains a certificate revocation list (CRL) to track the validity of certificates [18]. To prevent the need to download the CRL from the CA during authentication, each device carries the

CRL along with its certificate and exchanges both as its credentials. Given the global nature of the UCN, the number of entries in the CRL could be quite large resulting in CRLs that exceed the storage capacity of the APs and clients. For example, using `OpenSSL` each entry in a CRL adds 28 bytes to the size of the CRL. Assuming 10% revocation rate of one million users results in a CRL size of around 2.5MB. For comparison, the flash memory of the Linksys WRT54G is only 4MB. Therefore the CRL is partitioned and managed in segments. Every certificate falls in a subspace for that certified name encoded by a CRL segment called the "containing CRL". A certificate's non-existence in its containing CRL proves the validity of the certificate up to the time when the CRL segment was issued.

Assigning the containing CRL for a given participant is an important task to reduce the number of CRL segment lookups and the corresponding network overhead. If an authenticating client has the same containing CRL as the AP (a local authentication) then no lookup is required. Therefore, the assignment of participants to CRL segments should be done to maximize the number of local authentications. Users typically spend most of their time in a small number of physical locations (at work, at home, at a favorite cafe). Thus the majority of authentications can be expedited if devices in the same physical locations share the same containing CRL. To map locations to CRL segments the physical areas are divided into non-overlapping zones which completely cover the area. Each zone is represented by one CRL segment which is assigned a unique id. All participants located in the same zone have their certificates assigned to the same CRL segment.

Any of the existing solutions for application layer multicast, publish subscribe, or dissemination overlays could be used [65]. With the members organized into overlays based on the containing CRL (shared node id subspace) it is easy for the certificate authority to publish new updates. The CA can use the p2p overlay to find multiple members in the destination region and then publish the update to those nodes. When the new update is received it is distributed to all nodes according to the overlay specifics. Upon joining, new nodes are presented the most recent updates. The overlays could be optimized using coordination information from the bootstrap server and taking advantage of the embedded location information needed for assignment of node ids to form geographic overlays.

7.3.1 P2P CRL Segment Look-up Overlay

In most cases devices can verify each other using the locally stored certificates and CRL segments. Occasionally, however, the presented CRL segment might be deemed too old for the AP's freshness criteria. In a global heterogeneous system of autonomous entities like the UCN, it is difficult to reach consensus regarding a freshness cutoff. Rather than enforce a single threshold, AGE APs are organized into

a peer-to-peer overlay to find the most recent version of a CRL segment. A p2p network is favored over centralized approaches like OCSP [104] because of its resiliency and zero infrastructure cost. However, if a central server was deployed and available, it could also be used. To reduce network load, the CA maintains and publishes CRL segment timestamps signed by the CA confirming the last update time for the segment. In order to verify the freshness of a CRL segment, only the timestamp need be transferred.

One important note is that the CRL segment freshness look-up through the social network overlay service compromises the locality of the AGE authentication and therefore should be avoided whenever possible. Indeed, we envision that such remote checks will rarely be needed and only occur when very high confidence regarding the validity of a certificate is desired or a fraudulence attack suspected.

In order to verify the freshness of a given CRL segment the AGE p2p network must provide a trustworthy lookup function to find the timestamp for a given CRL id. Although distributed hash tables [64, 111, 124] solve this problem, the lookup can be manipulated by any node in the routing path making the results untrustworthy. To avoid untrusted links the access points in AGE instead form a social network overlay based on the trusted friendship of the owners. Social networks were originally studied under the small world phenomenon by Kleinberg [83] who identified the optimal properties of small world graphs to enable efficient message routing. Social networks have since been used in several projects because of the trusted nature of friendship based routing [98, 99]. The SPROUT project [99] uses social connections from the Friendster online social web site to augment a traditional DHT. The social links provide increased trust in the authenticity of the routing path. The results show improved query hit ratio and reduced delay in the face of malicious nodes. AGE also leverages the increased trustworthiness of social links to provide DHT-like timestamp queries for verifying CRL segments.

The CRL segment query is unique in that the query needs to be forwarded only to an area, not a specific AP. Any AP with a containing CRL that matches the request can reply. Routing through the social network then is a matter of finding a friend or a friend of a friend, etc. who is in the target zone. Using the intuition that a friend closer to the target zone is more likely to know someone in the target zone, greedily forwarding the request to the friend closest to the target gives the best chance to minimize the routing hops.

The AGE social network uses the location of each zone to forward messages between friends. Each AP keeps a listing signed by the AGE CA of all known zones and their coordinates. During routing, an AP's location is the same as its containing zone. This location based approach to routing is the same as greedy-face forwarding in ad hoc and sensor wireless networks, with the addition of long distance links. Many solutions to this problem have been proposed in previous research [36, 80, 89, 91]. In these routing protocols

Zone	Members	Zone	Members	Zone	Members
1	11579	9	102333	16	10318
2	8635	10	41310	17	2718
3	39127	11	12236	18	131364
4	25348	12	71980	19	106329
5	420962	13	265402	20	13154
6	238867	14	562028	21	47395
7	102302	15	730577	22	264036
8	96880				

Table 7.2: Members per zone in SMS traces.

a node greedily forwards a message to the neighbor closest to the destination. If the current node is closest amongst its neighbors to the destination the algorithm switches to some variant of face routing until greedy forwarding can be resumed. However, recent work has shown that not all variants of face routing succeed when routing in arbitrary undirected planar network graphs [58]. Based on this result we use the GFG algorithm [36] to perform routing in the AGE social network.

The basic GFG routing will fail in AGE in the situation that an AP only has links to other APs in the same zone. To resolve the issue, AGE uses *intra*-zone connectivity of its friends to select the best next hop. Selecting the friend with the most intra-zone connections ensures routing can resume and increases the likelihood of finding a long distance link toward the destination. In the extreme case that an AP has no intra-zone friends and no friends with intra-zone friends the message is forwarded to a randomly selected friend.

When a new participant joins the UCN network she first registers with the UCN provider either through a dedicated registration server or using off-line methods. A unique username and temporary password are granted. She then installs the AGE software on her AP and client devices. A special registration application on the AP generates a new private/public key pair. Using the temporary password the public key is submitted to the AGE CA and a new certificate binding the public key to the new username is generated and transferred to the AP. The participant's client devices synchronize with the AP to get new certificates and CRL segments. Any future updates to the CRL segment or user certificate are pushed from the CA server to individual APs.

7.4 SMS Trace Analysis

The strength of AGE's p2p fallback look-up network depends on the connectivity of the social network upon which it is built. If the AP is not connected to the target zone through its social links then it will not be able to verify the client and will reject a valid client. To verify that social networks are well connected SMS traces from a large nationwide cellular provider were analyzed. The traces cover the individ-

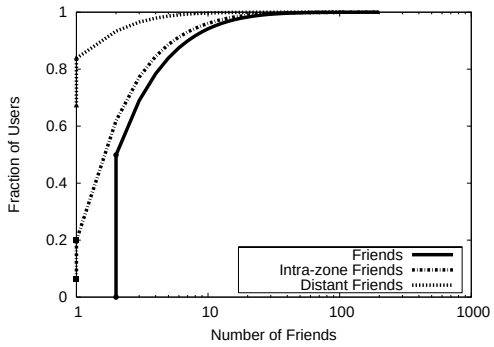


Figure 7.4: CDF of friends for each user including the total number of friends, the number of intra-zone friends and the number of distant friends. The fraction of users with 0 of each type of friend is marked by the lowest points on the line. Only users with between 2 and 200 friends were considered.

ual SMS messages from over 6 million users. In the traces two users are considered friends if they have exchanged at least one SMS message. Cellular networks provide a natural method for dividing users into zones. The processing of SMS messages is handled by the mobile switching center (MSC) in a hierarchical fashion so that all messages sent from a group of cell towers in a given area are forwarded to a single MSC. Overall in the trace there are 22 MSCs. The home MSC of any user can be extracted from his telephone number prefix. Each user is assigned to a zone based on his MSC. Using this breakdown the zone populations are as shown in Table 7.2.

The friendships of each user were first analyzed. One third of the users have only one friend, likely infrequent SMS users. In contrast, some users have tens of thousands of friends. These users likely are bots or messaging services from the cellular provider. Over the entire data set the mean, min and max number of friends is 2.3, 1, and 46789. Over 99.99% of users have fewer than 200 friends. Filtering out the outliers with fewer than 2 friends (infrequent users) and more than 200 friends (bots) the mean, min and max change to 4.2, 2, and 200 friends. Figure 7.4 shows the number of friends, number of intra-zone friends and number of long distance friends using the filtered user set. Intra-zone friends are those in different zones and long distant friends are those not in immediately adjacent zones. The GPS coordinates of each MSC were used to determine which zones are adjacent. Using the filtered user set, overall 94% of all users had a friend in a different zone. 33% of users have at least one long distance link.

To decide if these connectivity properties could support the forwarding of CRL segment lookups to all zones the reachability from every user to each of the 22 MSCs was evaluated. An MSC is reachable if any member in the MSC can be found following only the friendship links in the SMS social graph. The filtered set of users from above was used and the graph formed by friendship links of each user was followed, searching for each zone. Figure 7.5 shows how many zones were reachable by individual users. The analysis reveals the distribution of reachable MSCs (zones) is bimodal - 70% of users can reach all 22 MSCs

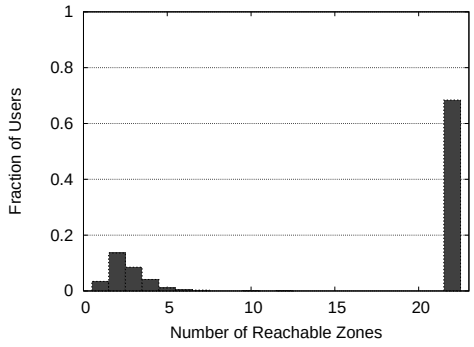


Figure 7.5: The maximum number of zones reachable by fractions of the population following only social links in SMS traces. The total number of zones is 22.

	EAP-TLS/LAN	EAP-TLS/UCN	EAP-AGE/UCN
Client Processing Time	0.021556	0.021171 [0.98x]	0.028369 [1.32x]
Server Processing Time	0.033561	0.013828 [0.41x]	0.485165 [14.46x]
Network Delay	0.065948	0.964469 [14.62x]	0.007132 [0.11x]
Total Auth. Delay	0.121065	0.999468 [8.26x]	0.520666 [4.30x]

Table 7.3: Latency breakdown in seconds for EAP-TLS over LAN (EAP-TLS/LAN), EAP-TLS over the Internet (EAP-TLS/UCN), and AGE over the Internet (AGE/UCN). The increase compared to the baseline (EAP-TLS/LAN) is shown in brackets for each measurement.

while the remaining can reach only 5 or less. Among the users with incomplete reachability the average number of friends per user was between 1.71 and 3.17. Among the well connected users the average number of friends was 5.02. This finding suggests that having at least 5 friends is sufficient to maintain the reachability to all zones in AGE's social network.

7.5 Experimental Results

The primary goals for this evaluation are to show that AGE is immune from Internet delays and achieves low authentication times. Authentication times of the AGE implementation were compared against industry standard implementations of other authentication methods in a realistic UCN deployment.

7.5.1 Implementation of AGE Framework

At its foundation AGE is based on EAP-TLS authentication [16], an EAP method using certificates and private keys. AGE is implemented as a new EAP type. New authentication modules for EAP-AGE were added to the popular Linux client supplicant software `wpa_supplicant` [15] and the open source RADIUS authentication server `FreeRADIUS` [6]. This implementation modifies the existing EAP-TLS methods by inserting the CRL exchange immediately following the EAP-Identity exchange. A background thread runs to validate CRLs, performing a lookup if necessary. The server configuration specifies the CRL toler-

	Avg.	Min.	Max.
EAP-TLS UCN	0.998509	0.987507	2.253010
EAP-AGE UCN	0.557370	0.497009	0.639851

Table 7.4: Variation in authentication times using normal EAP-TLS authentication compared to EAP-AGE.

ance to determine the acceptable age of a CRL. Using the `OpenSSL` library the CRL timestamp is checked. The validation thread also inserts a hook into the `OpenSSL` TLS engine for retrieving the CRL. When the EAP-TLS authentication reaches the point of CRL verification it calls the hook. The validating thread blocks until a reply is received from the social network. If the returned timestamp matches the timestamp of the client's CRL then the TLS handshake ensues. Otherwise the validation thread returns an error and the TLS handshake fails. The `FreeRADIUS` AGE module is configured by a new `age` section in the `eap.conf` configuration file. In total the access point software is around 3500 lines of code and the client `wpa_supplicant` module is only 500 lines of code. All of the access point software runs on the `OpenWRT` Linux distribution for the Linksys WRT54G AP. The AGE firmware image is 2.8MB which fits within the 4MB WRT54Gv4 onboard flash memory.

7.5.2 Performance Evaluation

Using the above AGE implementation, it is shown through measuring both network and processing latencies at all stages of the authentication process how AGE reduces the typical authentication delay. As a baseline AGE's performance is compared to a typical IEEE 802.1x EAP-TLS deployment where the client connects to a powerful authentication server over fast and reliable LAN links. For performance measurements AGE is also compared to a direct application of IEEE 802.1x using EAP-TLS in a user-created network where the authentication server is reachable over the Internet (see Figure 7.2). An access point was installed on a campus in Europe configured to run either EAP-TLS or EAP-AGE. A Pentium 4 laptop equipped with an internal 802.11b/g card was used to perform a series of authentications using `wpa_supplicant`. In the EAP-TLS over LAN case the access point forwarded authentication requests to a second laptop in the same campus, configured to run the `FreeRADIUS` server. In the EAP-TLS over UCN scenario the requests were forwarded to a Pentium 4 workstation in the United States to perform RADIUS authentication. Finally the local access point was configured to perform EAP-AGE as in Figure 7.3. The CRL tolerance was set low so that no social network lookup was required.

Table 7.3 shows the breakdown of authentication costs for each measured deployment, using EAP-TLS over LAN, UCN and EAP-AGE over UCN. The server side processing latency for AGE at the access point is significantly higher than at the Pentium-4 PC because the access point is equipped with a 200MHz microcontroller. There is also a slight increase in the client side processing time even though the client ma-

Authentication Type	Total Avg. Delay
AGE (decentralized)	520.67ms
Location (centralized):	
City	555.06ms
Regional	501.18ms
North America	1345.60ms
South America	852.63ms
Europe	1756.98ms
Mideast	1657.46ms
Asia	5902.86ms

Figure 7.6: Comparison of authentication times in milliseconds between decentralized authentication using AGE and centralized authentication in PlanetLab.

chine remained the same in all measurements. This comes from the extra overhead of transferring and processing the CRL segment. However, compared to EAP-TLS over UCN, AGE cuts the overall total authentication latency by half. The savings comes from the reduced network overhead, which is an order of magnitude lower than even the EAP-TLS over LAN configuration. Note that 94.3% of the AGE latency actually comes from the processing latency at the access point, which can be easily further reduced by a faster microcontroller. From Table 7.4 it is clear that the EAP-AGE protocol reduces jitter in authentication times offering more stability for sensitive applications like VoIP.

To further quantify the savings from using AGE, a second set of authentication delay measurements was taken from a residential setting. Over a residential broadband connection compared the authentication latency was measured using AGE versus TLS authentication to centralized servers throughout PlanetLab. The delays in PlanetLab should be comparable to the delays of tunneling to authentication servers in the Internet. Figure 7.6 shows average authentication times to locations around the world and the average AGE authentication time. Again, the weak processing power of the AP is the biggest constraint for AGE which none-the-less out-performs the centralized version in nearly all scenarios. Because AGE authentication is not dependent on network communication, the latency can be bound and easily improved as hardware speeds increase.

7.6 Future Work

One issue not addressed in this chapter is how to effectively ensure that UCN clients are updated with the most recent CRL segments and certificates. There may be long periods of time when the client does not associate to its home AP, for example during a business trip, or holiday. During this time the client's credentials may expire. If the client has invalid authentication material then it cannot authenticate and therefore cannot download the most recent material. One possibility is to open a port on the AGE access point which only allows connectivity to the CA for updating authentication material. At the same time,

if the CA is unreachable the client may still fail to update its CRL. Another possibility is to automatically look-up the CRL segment in the social overlay, but this approach shifts the burden of proof from the client onto the AP and potentially opens the door to denial of service attacks.

An important optimization for the UCN is support for fast hand-offs between UCN access points. Most existing solutions rely on a proxy authenticator to handle fast rekeying or pre-authentication by the client at neighboring access points. UCN's unmanaged infrastructure makes such approaches difficult or impossible. Future research should examine how to best support fast re-authentication in UCN networks, potentially utilizing static wireless connections with other gateways to facilitate the hand-offs.

7.7 Conclusion

Authentication on the Edge (AGE) running in UCNs will pave the way for wide scale adoption of UCN, both from the AP owner and ISP operator perspectives. Users can trust the security and performance enabled by AGE while operators can be confident in having the traceability and accountability they require. AGE's mechanisms of localizing and distributing authentication enable security and accountability to easily scale with the enormous growth potential inherent in UCNs. Combined with other network management presented in this thesis and future work, AGE will open the door for inter-network communication between the UCN and the Internet.

Chapter 8

Future Work

This thesis has focused on networking services to enable both intra- and inter-network communication in user-created networks. One issue not addressed is how to bridge the two styles of communication. Each makes very different assumptions about the network environment and node availability. In addition, before the UCN will be widely used, a number of other services will also need to be developed. Perhaps the biggest obstacle for the UCN once communication has been enabled is to motivate participation in the network. New business models need to be developed that allow existing network operators to benefit financially from the network and not immediately squash the UCN. End-users' data and network security must be secured and incentive schemes deployed that encourage users to share their private resources. Of course, the unique environment of the UCN prevents existing approaches for these problems from being directly deployed in the UCN.

8.1 Bridging Routing Protocols

In this work, intra- and inter-network communication were treated as completely independent modes of operation for UCN devices. In reality, as devices move throughout the network, the most appropriate mode of operation changes depending on the current situation. In fact, as the density of nodes fluctuates in a given area, the connectivity between devices varies from only intermittently-connected to traditional ad hoc connectivity to more static mesh environments to traditional fixed Internet connectivity. One challenge is to monitor the current conditions and dynamically chose the best networking approach based on the given connectivity and application demands.

The primary challenges to building such a framework include i) finding network metrics that accurately indicate the connectivity levels in the network and choosing routing approaches that work best for the given connectivity, ii) transitioning between routing protocols when the conditions change and iii) allowing enough flexibility to incorporate new routing protocols and designs as they are developed. Monitoring the network is hard because the choice of routing protocol depends on the end-to-end connectivity in

the network. Measurement tools such as iperf [107], traceroute [75] and pchar [96] generate high network overhead and require fixed endpoints for measurement. However, without regular measurement, it will be difficult to differentiate between short-term channel failures, route changes from mobility or network partitioning. Essentially, new measurement techniques are needed.

The granularity of measurement and protocol selection period will also have a big impact on the framework success. As the conditions at each node change, the framework will need to decide how to handle existing data flows. For example, if a node moves from an ad hoc scenario where it is sending TCP flows through the network, what happens to those flows when an intermittently-connected state is reached. Or, if using a pro-active ad hoc protocol, when should the node stop sending routing updates or switch to sending updates for a mesh or fixed routing protocol.

Finally, an ideal framework will be able to incorporate future routing protocols and network environments not currently imagined. Similar to the design of IP, which has proven successful in incorporating any type of underlying network, the routing framework should be able to easily include new routing protocols in the future. It is not clear if this can be done by creating a new network stack with the narrow waste above the routing layer instead of IP or if a different cross-layer approach is needed.

8.2 Economic and Legal Challenges

In this thesis, the economic challenges for UCNs have not been addressed. The majority of ISPs do not allow Internet access to be shared or resold. Existing providers are reluctant to commodify their infrastructure by allowing competition for service provisioning. Similarly, cellular providers make money from selling data access. If customers move into the UCN instead of using cellular services, operators can stand to lose customers and profits. However, the deployment of the UCN results in higher demand for broadband connectivity and improved coverage for existing cellular customers. If revenue sharing models can be agreed upon, both the end-users and existing providers can mutually benefit.

There is also a legal concern of ISPs allowing non-customers to access their network through a shared base station. An approach based on tunneling has been proposed for a citywide open access infrastructure [113]. While this approach addresses the legal concerns of both the users and the ISPs, the tunneling approach adopted sends all traffic back to the roaming customer's home base station and to his own broadband connection. This has several drawbacks in that it prevents users who do not have a broadband connection or who are not part of the network to use the base station, it limits the available bandwidth to the uplink bandwidth of the broadband connection, and it incurs high latency as the distance between

base stations grows making regional or national networks impracticable. Those issues would need to be addressed for deployment in the organic network.

The UCN must provide mechanisms to further support the enforcement of whatever policies ISPs design in the future. One important feature is the collection of metrics and network data for off-line analysis. This network data will provide the information ISPs need to craft their policies and ensure that they are being enforced. Such data collection is challenging in the widely distributed UCN with unreliable nodes and effective solutions will have to deal with the limited resources available at the gateways.

8.3 Incentive Management

Users in the UCN may not be willing to share their resources unless there is tangible benefit for them. The benefit can come in the form of reciprocal data access or financial incentives. Credit-based incentive schemes have been proposed for traffic forwarding in ad hoc networks [35, 138] but they require centralized brokers or trusted hardware for managing payments. Auction based systems similar to MOB [40] rely on an established monetary system that requires a centralized authority to issue money and manage payments. Reputation-based schemes on the other hand are decentralized and operate by monitoring neighbor behavior and in conjunction with other complying nodes punishing delinquents [68, 97]. The OCEAN system [32] uses only locally available information and a crediting system to overcome selfish nodes in ad hoc networks. The problem with strict crediting schemes or tit-for-tat approaches is that users in remote and isolated locations cannot earn credit and are unfairly punished for their location in the network.

One alternative approach can be borrowed from the electrical power grid [60]. In the power grid, customers can install generators on their own property and contribute power back to the grid. The customer is paid for his contribution as a credit against his bill or, if a surplus is generated, direct payment. In the same way the organic operator can buy the customer's resources as an incentive to the customer to participate. The challenging question is to define what resource is used to meter the customer's contribution. The obvious choice is to meter the data transferred through the customer's wireless network. However, such an approach disfavors customers in less popular areas who experience far less traffic at their base station. These customers are less likely to participate because the possible rewards are small, slowing expansion of the network. Instead, the customer might be rewarded for the amount of control or storage traffic that he contributes for the network services. While the second approach is location agnostic, it fails to directly promote wireless sharing. The final decision for user incentive will have a big impact on the financial success of the network and must be carefully considered.

Regardless of the incentive mechanisms deployed in the UCN, traffic shaping at the UCN gateways will be an important tool. By using AGE, clients can be uniquely identified. Using the client's certified identity, the gateway assigns traffic as either owner or foreign traffic. Traffic from the gateway owner always has preferential treatment. Foreign traffic uses whatever bandwidth is left over. If the incentive scheme allows it, foreign traffic can be further separated into different categories of service depending on the client's incentive status. Such traffic shaping techniques will be especially important to end-users who are charged for data access on a per-usage basis. In their case, strict traffic control is necessary to avoid exceeding their subscription limits.

8.4 Security and Privacy Policies

One of the reasons users might hesitate to join the PERM community is a fear over privacy and security attacks made possible in open Wi-Fi networks. A malicious neighbor might willingly share his Internet access in order to steal sensitive data sent across his gateway or in order to perform illegal activities on his neighbors' connections, such as downloading/uploading illegal media. Hidden behind his neighbor's ISP, it is difficult to determine who the actual perpetrator is. A malicious neighbor can easily sniff packets between his wireless access point and broadband modem without detection. Note that end-to-end encryption does not protect IP headers, which can reveal private information. For example, frequent access to a certain bank web site implies that it is the user's personal bank account. Any captured data can be more sensitive given that the attacker also knows the physical location of the victim and can combine the snooping attacks with social engineering attacks.

The best solution to prevent data theft is to tunnel any traffic over the neighbor's gateway to a secure proxy in the Internet. For example a TOR client[10] or VPN tunnel could be integrated with the PERM scheduler for foreign connections. There are two drawbacks to this approach that lead to the inclusion of additional mechanisms in the PERM design. First, in some cases a trusted proxy may not always be available. Second, there is a performance loss in terms of latency, possibly reduced bandwidth at the proxy and also loss of accuracy in the PERM scheduler's RTT estimations (especially in the case of TOR, which follows random paths to the destination).

Besides protecting outgoing traffic, the PERM system must also protect the local resources. All foreign clients should be unable to reach any machines on the local network or access the gateway configuration. To hinder clients from performing illegal activities while 'hiding' behind the owner's IP address, extra security policies are needed.

White and black lists can be used to filter traffic that should either not be sent across a foreign gateway or not accessed by foreign clients over the owner's gateway. For example, all access attempts to sensitive destinations, such as financial and medical institutes, adult web-sites, and governmental agencies, could be blocked based on their hierarchical DNS names and IP addresses. Individual users can configure the rules at their home wireless routers based on their own preferences. However, this approach demands a high level of knowledge and awareness from the end-user to configure and maintain his network. New automated techniques and configuration tools are needed to aid in this step.

Chapter 9

Conclusion

UCNs have the potential to quickly and cheaply improve the end-user mobile networking experience by harnessing the many idle resources in end-user devices. To build a communications foundation for these networks, new services must be deployed. Because the UCN is formed entirely of devices owned and operated by end-users, the network environment poses connectivity challenges for intra-network communication services and management challenges for inter-network communication services. The constraints of running on end-user devices mean existing centralized, global and static approaches are unable to operate in the UCN. In response, this thesis has presented new dynamic, decentralized and local protocols and algorithms for data overlays and authentication in UCNs. In addition to individual management of connectivity and existing data forwarding schemes, all of these services suffer from limited resources in the end-user devices. To enable high throughput, low latency and more reliable services, new resource management techniques have been presented in this thesis for each of the above services.

The performance of these services provides a strong argument for the viability and usefulness of the user-created network. This thesis has demonstrated that decentralized local and location-based algorithms are the perfect fit for a network formed by spontaneous, direct connections between end-users. As the awareness of the opportunities from user-created networks grows, both among end-users and operators, the participation in the networks will also grow, resulting in increased benefit. As demand and use grow, there will be a stronger need and priority to solve the remaining incentive and business challenges that still stand in the way of ubiquitous deployment of the UCN.

References

[1] Abitcool wi-fi community. http://www.abitcool.com/.

[2] Boingo wireless service. http://boingo.com/.

[3] Canadians leads the world in time spent on the internet. http://www.digitalhome.ca/content/view/1732/1/.

[4] Champaign-Urbana community wireless network. http://www.cuwireless.net/.

[5] Extended HotSpots. http://www.extendedhotspots.net/opencms/opencms/index.html?_locale=en.

[6] Freeradius. http://www.freeradius.org.

[7] Global wi-fi community. http://www.fon.com/.

[8] June 2007 bandwidth report. http://www.websiteoptimization.com/bw/0706/.

[9] MIT roofnet. http://www.pdos.lcs.mit.edu/roofnet/.

[10] The onion router. http://tor.eff.org.

[11] OpenWrt Linux. http://www.openwrt.org/.

[12] Rice transit access points. http://taps.rice.edu/.

[13] Seattle wireless. http://www.seattlewireless.net/.

[14] T-mobile hotspots. http://hotspot.t-mobile.com/.

[15] Wpa supplicant. http://hostap.epitest.fi/wpa_supplicant/.

[16] B. Aboba and D. Simon. PPP EAP TLS Authentication Protocol. RFC 2716 (Experimental), October 1999. Obsoleted by RFC 5216.

[17] Bernard Aboba. Virtual access points, May 2003. IEEE P802.11 Wireless LANs.

[18] C. Adams and S. Farrell. Internet X.509 Public Key Infrastructure Certificate Management Protocols. RFC 2510 (Proposed Standard), March 1999. Obsoleted by RFC 4210.

[19] Hari Adiseshu, Guru Parulkar, and George Varghese. A reliable and scalable striping protocol. In *Proceedings of ACM SIGCOMM*, August 1996.

[20] Atul Adya, Paramvir Bahl, and Lili Qiu. Analyzing the browse patterns of mobile clients. In *Proceedings of SIGCOMM Internet Measurement Workshop*, 2001.

[21] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. *SIGCOMM Comput. Commun. Rev.*, 34(4), 2004.

[22] Aditya Akella, Bruce Maggsy, Srinivasan Seshan, Anees Shaikh, and Ramesh Sitaraman. A measurement-based analysis of multihoming. In *Proceedings of ACM SIGCOMM*, 2003.

[23] Aditya Akella, Jeffrey Pang, Bruce Maggsy, Srinivasan Seshan, and Anees Shaikh. A comparison of overlay routing and multihoming route control. In *Proceedings of ACM SIGCOMM*, 2004.

[24] Aditya Akella, Srinivasan Seshan, and Anees Shaikh. Multihoming performance benefits: An experimental evaluation of practical enterprise strategies. In *Proceedings of USENIX Annual Technical Conference*, 2004.

[25] M. Allman, V. Paxson, and W. Stevens. TCP congestion control. RFC 2581, Internet Engineering Task Force, April 1999.

[26] Ganesh Ananthanarayanan, Venkata N. Padmanabhan, Lenin Ravindranath, and Chandramohan A. Thekkath. Combine: Leveraging the power of wireless peers through collaborative downloading. In *Proceedings of ACM MobiSys*, 2007.

[27] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Robert Morris. Resilient overlay networks. In *Proceedings of ACM SOSP*, 2001.

[28] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Rohit Rao. Improving web availability for clients with MONET. In *Proceedings of NSDI*, 2005.

[29] Yossi Azar, Andrei Z. Broder, and Anna R. Karlin. On-line load balancing. In *Proceedings of ACM Symposium on Foundations of Computer Science*, 1992.

[30] Aruna Balasubramanian, Brian Neil Levine, and Arun Venkataramani. DTN routing as a resource allocation problem. In *Proc. ACM SIGCOMM*, 2007.

[31] Nilanjan Banerjee, Mark D. Corner, and Brian Neil Levine. Design and field experimentation of an energy-efficient architecture for dtn throwboxes. *IEEE/ACM Trans. Netw.*, 18(2):554–567, 2010.

[32] Sorav Bansal and Mary Baker. Observation-based cooperation enforcement in ad hoc networks. http://arxiv.org/pdf/cs.NI/0307012, July 2003.

[33] R. Bhagwan, S. Savage, and G. Voelker. Understanding availability. In *Proceedings IPTPS*, 2003.

[34] Maria Bina and George Giaglis. Emerging issues in researching community-based wlans. *Journal of Computer Information Systems*, Fall 2005.

[35] Ljubica Blazevic, Levente Buttyan, Srdan Capkun, Silvia Giordano, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Self-organization in mobile ad-hoc networks: the approach of terminodes. *IEEE Communications Magazine*, 39(6):166–174, June 2001.

[36] Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.

[37] John Burgess, Brian Gallagher, David Jensen, and Brian Neil Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *IEEE Infocomm*, 2006.

[38] I. Castineyra, N. Chiappa, and M. Steenstrup. The nimrod routing architecture. RFC 1992, August 1996.

[39] V. Cerf, Y. Dalal, and C. Sunshine. Specification of Internet Transmission Control Program. RFC 675, December 1974.

[40] Rajiv Chakravorty, Suman Banerjee, Sulabh Agarwal, and Ian Pratt. MoB: A mobile bazaar for wide-area wireless services. In *Proceedings of ACM MobiCom*, 2005.

[41] Ranveer Chandra, Paramvir Bahl, and Pradeep Bahl. MultiNet: Connecting to multiple IEEE 802.11 networks using a single wireless card. In *Proceedings of IEEE INFOCOM*, 2004.

[42] Kameswari Chebrolu and Ramesh Rao. Communication using multiple wireless interfaces. In *Proceedings of IEEE WCNC*, March 2002.

[43] Francisco Chinchilla, Mark Lindsey, and Maria Papadopouli. Analysis of wireless information locality and association patterns in a campus. In *Proceedings of IEEE INFOCOM*, 2004.

[44] Ionut Constandache, Romit Roy Choudhury, and Injong Rhee. Towards mobile phone localization without war-driving. In *Proceedings IEEE Infocomm*, 2010.

[45] CoreStreet. Distributed certificate validation. http://www.corestreet.com/about/library/whitepapers/w06-04v2-distributed_certificate_validation.pdf.

[46] Paolo Costa, Davide Frey, Matteo Migliavacca, and Luca Mottola. Towards lightweight information dissemination in inter-vehicular networks. In *VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, pages 20–29, New York, NY, USA, 2006. ACM.

[47] Samir Ranjan Das, Charles E. Perkins, and Elizabeth M. Belding-Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proceedings IEEE INFOCOM*, 2000.

[48] Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. In *ACM PODC*, 1987.

[49] Marios D. Dikaiakos, Saif Iqbal, Tamer Nadeem, and Liviu Iftode. VITP: an information transfer protocol for vehicular computing. In *International Conference on Mobile Computing and Networking*, 2005.

[50] Einar Edvardsen. Fixed and mobile convergence. In *Proceedings of BroadBand Europe*, 2004.

[51] Elias C. Efstathiou, Pantelis A Frangoudis, and George C Polyzos. Stimulating participation in wireless community networks. In *Proceedings of IEEE INFOCOM*, 2006.

[52] J. Eriksson and S. Madden. Cabernet: vehicular content delivery using WiFi. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 199–210. ACM New York, NY, USA, 2008.

[53] Vijay Erramilli, Mark Crovella, Augustin Chaintreau, and Christophe Diot. Delegation forwarding. In *ACM MobiHoc*, 2008.

[54] Randall Stewart et al. Stream control transmission protocol. RFC 2960, October 2000.

[55] Kevin Fall. A delay-tolerant network architecture for challenged Internets. In *Proceedings of ACM SIGCOMM*, 2003.

[56] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.*, 1(4):397–413, 1993.

[57] Walter Franz, Hannes Hartenstein, and Bernd Bochow. Internet on the Road via Inter-Vehicle Communications. In *GI Workshop: Communications over Wireless LANs'*, 2001.

[58] Hannes Frey and Ivan Stojmenovic. On delivery guarantees of face and combined greedy-face routing in ad hoc and sensor networks. In *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, 2006.

[59] Sean Gallagher. At&t wants you off its network, iphone user! http://www.techgoesstrong.com/att-wants-you-its-network-iphone-user.

[60] Jeffrey Gangemi. Selling power back to the grid. Business Week Online (http://tinyurl.com/3crts5).

[61] Yuan Gao, Guanghui He, and Jennifer C. Hou. On exploiting traffic predictability in active queue management. In *Proceedings of IEEE INFOCOM*, 2002.

[62] Krishna P. Gummadi, Harsha Madhyastha, Steven D. Gribble, Henry M. Levy, and David J. Wetherall. Improving the reliability of internet paths with one-hop source routing. In *Proceedings of USENIX OSDI*, 2004.

[63] Fanglu Guo, Jiawu Chen, Wei Li, and Tzi-cker Chiueh. Experiences in building a multihoming load balancing system. In *Proceedings of IEEE INFOCOM*, 2004.

[64] Indranil Gupta, Ken Birman, Prakash Linga, Al Demers, and Robbert van Renesse. Kelips: Building an efficient and stable P2P DHT through increased memory and background overhead. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, 2003.

[65] Indranil Gupta, Anne-Marie Kermarrec, and Ayalvadi J. Ganesh. Efficient epidemic-style protocols for reliable and scalable multicast. *Reliable Distributed Systems, IEEE Symposium on*, 0:180, 2002.

[66] Piyush Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, IT-46(2):388–404, Mar. 2000.

[67] T. Hara. Effective replica allocation in ad hoc networks for improving data accessibility. *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 3:1568–1576 vol.3, 2001.

[68] Qi He, Dapeng Wu, and Pradeep Khosla. Sori: A secure and objective reputation-based incentive scheme for ad-hoc networks. In *Proceedings of IEEE WCNC*, 2004.

[69] Tristan Henderson, David Kotz, and Ilya Abyzov. The changing usage of a mature campuswide wireless network. In *Proceedings of ACM MobiCom*, 2004.

[70] Hung-Yung Hsieh and Raghupathy Sivakumar. A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts. In *Proceedings of ACM MobiCom*, 2002.

[71] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 241–250, New York, NY, USA, 2008. ACM.

[72] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. CarTel: a distributed mobile sensor computing system. *Conference On Embedded Networked Sensor Systems*, 2006.

[73] J. Hrri, M. Fiore, F. Fethi, and C. Bonnet. Vanetmobisim: generating realistic mobility patterns for vanets. In *In Proc. of the 3rd ACM International Workshop on Vehicular Ad Hoc Networks (VANET'06)*, 2006.

[74] Khaled Ibrahim, Michele C. Weigle, and Mahmoud Abuelela. p-IVG: Probabilistic Inter-Vehicle Geocast for Dense Vehicular Networks. *VTC Spring 2009 - IEEE 69th Vehicular Technology Conference*, pages 1–5, April 2009.

[75] Van Jacobson. traceroute.

[76] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *CoNEXT '09: Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12, New York, NY, USA, 2009. ACM.

[77] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *In Proceedings of the Fifth ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT 2009)*, 2009.

[78] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. In *Proceedings of ACM SIGCOMM*, 2004.

[79] David B Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, volume 353, pages 153–181. Kluwer Academic Publishers, 1996.

[80] Brad Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of ACM MobiCom*, 2000.

[81] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumor spreading. In *IEEE FOCS*, 2000.

[82] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. The one simulator for dtn protocol evaluation. In *Simutools*, 2009.

[83] Jon Kleinberg. The small-world phenomenon: an algorithm perspective. In *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 163–170, New York, NY, USA, 2000. ACM Press.

[84] Paul Kocher. A quick introduction to certificate revocation trees (crts). In *Technical report, ValiCert*, 1990.

[85] E. Kohler, M. Handley, and S. Floyd. Datagram Congestion Control Protocol (DCCP). RFC 4340 (Proposed Standard), March 2006.

[86] David Kotz and Kobby Essien. Analysis of a campus-wide wireless network. In *Proceedings of ACM MobiCom*, 2002.

[87] David Kotz, Tristan Henderson, and Ilya Abyzov. CRAWDAD data set dartmouth/campus (v. 2004-12-18). Downloaded from http://crawdad.cs.dartmouth.edu/dartmouth/campus, December 2004.

[88] P. Kouznetsov, R. Guerraoui, S.B. Handurukande, and A.-M. Kermarrec. Reducing noise in gossip-based reliable broadcast. *Proceedings IEEE Symposium on Reliable Distributed Systems*, 2001.

[89] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Worst-case optimal and average-case efficient geometric ad-hoc routing. In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, 2003.

[90] Manish Lad, Saleem Bhatti, Steve Hailes, and Peter Kirstein. Coalition-based peering for flexible connectivity. In *Proceedings PIMRC*, 2006.

[91] B. Leong, S. Mitra, and B. Liskov. Path vector face routing: Geographic routing with local face information. In *Proceedings of ICNP 2005*, 2005.

[92] Philip Levis, Neil Patel, David Culler, and Scott Shenker. Trickle: a self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings USENIX NSDI*, 2004.

[93] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, 2003.

[94] K. Lougheed and Y. Rekhter. Border Gateway Protocol (BGP). RFC 1105 (Experimental), June 1989. Obsoleted by RFC 1163.

[95] Luiz Magalhaes and Robin Kravets. Transport level mechanisms for bandwidth aggregation on mobile hosts. In *Proceedings of IEEE ICNP*, 2001.

[96] Bruce A. Mah. pchar: A tool for measuring internet path characteristics.

[97] Ratul Mahajan, Maya Rodrig, David Wetherall, and John Zahorjan. Sustaining cooperation in multi-hop wireless networks. In *Proceedings of NSDI*, 2005.

[98] G. Manku, M. Bawa, and P. Raghavan. Symphony: Distributed hashing in a small world. In *Proceedings of USITS 2003*, 2003.

[99] Sergio Marti, Prasanna Ganesan, and Hector Garcia-Molina. Sprout: P2p routing with social networks. In *Proceedings of First International Workshop on Peer-to-Peer and Databases (P2PDB 2004)*, 2004.

[100] Marco Mellia, Ion Stoica, and Hui Zhang. Tcp model for short lived flows, Februrary 2002.

[101] Silvio Micali. Efficient certificate revocation. In *Technical Memo MIT/LCS/TM-542b*, 1996.

[102] Allen Miu, Godfrey Tan, Hari Balakrishnan, and John Apostolopoulos. Divert: Fine-grained path selection for wireless lans. In *Proceedings of ACM MobiSys*, 2004.

[103] J. Moy. OSPF version 2, RFC 2178. `http://www.ietf.org/rfc/rfc2178.txt`.

[104] M Myers, R Ankney, A Malpani, S Galperin, and C Adams. Online certificate status protocol - OCSP. IETF Request For Comments (RFC2560), 1999.

[105] Samuel C. Nelson, Mehedi Bakht, and Robin Kravets. Encounter-based routing in dtns. In *Proc. IEEE Infocomm*, 2009.

[106] Samuel C. Nelson, Albert F. Harris III, and Robin Kravets. Event-driven, role-based mobility in disaster recovery networks. In *ACM CHANTS*, 2007.

[107] NLANR. Iperf network profiling tool.

[108] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose. Modeling TCP throughput: a simple model and its empirical validation. In *Proceedings of ACM SIGCOMM*, 1998.

[109] Charles E. Perkins and Elizabeth M. Royer. Ad-hoc on demand distance vector routing. In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[110] Michal Piórkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. On clustering phenomenon in mobile partitioned networks. In *MobilityModels '08: Proceeding of the 1st ACM SIGMOBILE workshop on Mobility models*, pages 1–8, New York, NY, USA, 2008. ACM.

[111] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Schenker. A scalable content-addressable network. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, 2001.

[112] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for largescale peer-to-peer systems. In *Proceedings of Conference on Distributed Systems Platforms, Heidelberg*, 2001.

[113] Nishanth Sastry, Jon Crowcroft, and Karen Sollins. Architecting citywide ubiquitous wi-fi access. In *Proc. ACM HotNets*, 2007.

[114] Stefan Savage, Thomas Anderson, Amit Aggarwal, David Becker, Neal Cardwell, Andy Collins, Eric Hoffman, John Snell, Amin Vahdat, Geoff Voelker, and John Zahorjan. Detour: Informed internet routing and transport. *IEEE Micro*, 19(1):50–59, January/February 1999.

[115] Puneet Sharma, Sung-Ju Lee, Jack Brassil, and Kang G. Shin. Handheld routers: Intelligent bandwidth aggregation for mobile collaborative communities. In *Proceedings of IEEE BROADNETS*, 2004.

[116] M. Shinohara, T. Hara, and S. Nishio. Data Replication Considering Power Consumption in Ad Hoc Networks. *Mobile Data Management, 2007 International Conference on*, pages 118–125, 2007.

[117] R. Shorten, F. Wirth, and D. Leith. A positive systems model of tcp-like congestion control: Asympototic results. Technical Report 2004-1, Hamilton Institute, April 2004.

[118] Tara Small and Zygmunt J. Haas. The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In *ACM MobiHoc*, 2003.

[119] Tara Small and Zygmunt J. Haas. Resource and performance tradeoffs in delay-tolerant wireless networks. In *SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[120] Alex Snoeren. Adaptive inverse multiplexing for widearea wireless networks. In *Proceedings of IEEE GLOBECOM*, 1999.

[121] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[122] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. In *PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 79–85, Washington, DC, USA, 2007. IEEE Computer Society.

[123] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Efficient routing in intermittently connected mobile networks: the multiple-copy case. *IEEE/ACM Trans. Netw.*, 16(1):77–90, 2008.

[124] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup protocol for Internet applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, February 2003.

[125] Ion Stoica, Hui Zhang, and T. S. Eugene Ng. A hierarchical fair service curve algorithm for link-sharing, real-time and priority services. *IEEE/ACM Transactions on Networking*, 8(2):185–199, April 2000.

[126] Bin Tang, Himanshu Gupta, and Samir R. Das. Benefit-based data caching in ad hoc networks. *Mobile Computing, IEEE Transactions on*, 7(3):289–304, March 2008.

[127] Shu Tao, Kuai Xu, Ying Xu, Teng Fei, Lixin Gao, Roch Guerin, Jim Kurose, Don Towsley, and Zhi-Li Zhang. Exploring the performance benefits of end-to-end path switching. In *Proceedings of IEEE ICNP*, 2004.

[128] Amin Vahdat and David Becker. Epidemic routing for partially-connected ad hoc networks. Technical Report CS-200006, Duke University, April 2000.

[129] L. Wischoff, A. Ebner, H. Rohling, M. Lott, and R. Halfmann. Sotis - a self-organizing traffic information system. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, pages 2442–2446 vol.4, April 2003.

[130] Hao Wu, Richard Fujimoto, Randall Guensler, and Michael Hunter. MDDV: a mobility-centric data dissemination algorithm for vehicular networks. In *ANET '05: Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, 2004.

[131] Lily Yang, Petros Zerfos, and Emek Sadot. Architecture taxonomy for control and provisioning of wireless access points (CAPWAP). IETF Request For Comments (RFC4118), 2005.

[132] Liangzhong Yin and Guohong Cao. Supporting cooperative caching in ad hoc networks. *Mobile Computing, IEEE Transactions on*, 5(1):77–89, Jan. 2006.

[133] Xiaolan Zhang, Jim Kurose, Brian Neil Levine, Don Towsley, and Honggang Zhang. Study of a bus-based disruption-tolerant network: mobility modeling and impact on routing. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 195–206, New York, NY, USA, 2007. ACM.

[134] Xiaolan Zhang, Giovanni Neglia, Jim Kurose, and Don Towsley. Performance modeling of epidemic routing. *Computer Networks*, 51(10):2867–2891, 2007.

[135] Y. Zhang, B. Hull, H. Balakrishnan, and S. Madden. ICEDB: Intermittently-Connected Continuous Query Processing. *International Conference on Data Engineering (ICDE), Istanbul, Turkey*, pages 166–175, 2007.

[136] Jing Zhao and Guohong Cao. VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, 57(3):1910–1922, 2008.

[137] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc '04: Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 187–198, New York, NY, USA, 2004. ACM.

[138] Sheng Zhong, Jiang Chen, and Yang Richard Yang. Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *Proceedings of IEEE INFOCOM*, 2003.